ISSN 0265-2919

R 80p 2

THE HOME COMPUTER ADVANCED COURSE

MAKING THE MOST OF YOUR MICRO

An ORBIS Publication

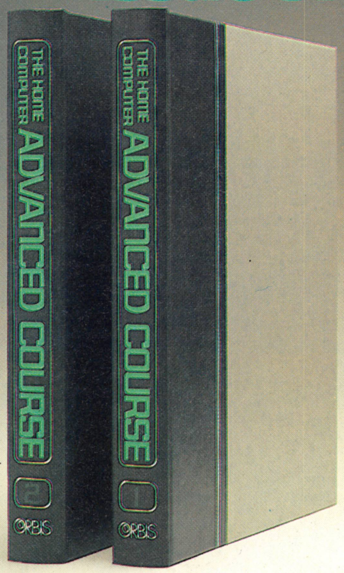
IR £1 Aus $1.95 NZ $2.25 SA R1.95 Sing $4.50 USA & Can $1.95

CONTENTS

Next Week

- **The BBC Microcomputer is now widely used throughout the educational system. We look at the strengths and weaknesses of BBC BASIC.**
- **Games programs are the bread and butter of the software industry. Games generators provide the means by which you can make the rules. With only the game framework supplied, the limiting factor is imagination.**
- **Machine code is the closest we can come to direct communication with the microcomputer. For program efficiency it can't be bettered.**

Your special offer binder order form will be with Issue 5.



THE HOME COMPUTER ADVANCED COURSE
2
ORBIS
THE HOME COMPUTER ADVANCED COURSE
1
ORBIS

Overseas readers: this special offer applies to readers in the U.K., Eire and Australia only.

COVER PHOTOGRAPHY BY CHRIS STEVENS

Editor Jonathan Hilton; **Consultant Editors** Gareth Jefferson, Richard Pawson; **Art Director** David Whelan; **Deputy Editor** Roger Ford; **Production Editor** Catherine Cardwell; **Staff Writer** Brian Morris; **Picture Editor** Claudia Zeff; **Designer** Hazel Bennington; **Sub Editors** Robert Pickering, Keith Parish; **Art Assistant** Liz Dixon; **Editorial Assistant** Stephen Malone; **Researcher** Melanie Davis; **Contributors** Lisa Kelly, Steven Colwill, Richard King, Martin Hayman; **Group Art Director** Perry Neville; **Managing Director** Stephen England; **Published by Orbis Publishing Ltd: Editorial Director** Brian Innes; **Project Development** Peter Brookesmith; **Executive Editor** Chris Cooper; **Production Co-ordinator** Ian Paton; **Circulation Director** David Breed; **Marketing Director** Michael Joyce; **Designed and produced by Bunch Partworks Ltd; Editorial Office** 85 Charlotte Street, London W1; **© 1984 by Orbis Publishing Ltd: Typeset by Universe; Reproduction by Mullis Morgan Ltd; Printed in Great Britain by Artisan Press Ltd, Leicester**

HOME COMPUTER ADVANCED COURSE – Price UK 80p IR £1.00 AUS $1.95 NZ $2.25 SA R1.95 SINGAPORE $4.50 USA and CANADA $1.95
How to obtain your copies of HOME COMPUTER ADVANCED COURSE – Copies are obtainable by placing a regular order at your newsagent, or by taking out a subscription. Subscription rates: for six months (26 issues) £23.80; for one year (52 issues) £47.60. Send your order and remittance to Punch Subscription Services, Watling Street, Bletchley, Milton Keynes, Bucks MK2 2BW, being sure to state the number of the first issue required.
Back Numbers UK and Eire – Back numbers are obtainable from your newsagent or from HOME COMPUTER ADVANCED COURSE. Back numbers, Orbis Publishing Limited, 20/22 Bedfordbury, LONDON WC2N 4BT at cover price. AUSTRALIA: Back numbers are obtainable from HOME COMPUTER ADVANCED COURSE. Back numbers, Gordon & Gotch (Aus) Ltd, 114 William Street, PO Box 767G Melbourne, Vic 3001. SOUTH AFRICA, NEW ZEALAND, EUROPE & MALTA: Back numbers are available at cover price from your newsagent. In case of difficulty write to the address in your country given for binders. South African readers should add sales tax.
How to obtain binders for HOME COMPUTER ADVANCED COURSE – UK and Eire: Details of how to obtain your binders (and of our special offer) are in issue 5. EUROPE: Write with remittance of £5.00 per binder (incl. p&p) payable to Orbis Publishing Limited, 20/22 Bedfordbury, LONDON WC2N 4BT. MALTA: Binders are obtainable through your local newsagent price £3.95. In case of difficulty write to HOME COMPUTER ADVANCED COURSE BINDERS, Miller (Malta) Ltd, M.A. Vassalli Street, Valletta, Malta. AUSTRALIA: For details of how to obtain your binders see inserts in early issues or write to HOME COMPUTER ADVANCED COURSE BINDERS, First Post Pty Ltd, 23 Chandos Street, St. Leonards, NSW 2065. The binders supplied are those illustrated in the magazine. NEW ZEALAND: Binders are available through your local newsagent or from HOME COMPUTER ADVANCED COURSE BINDERS, Gordon & Gotch (NZ) Ltd, PO Box 1595, Wellington. SOUTH AFRICA: Binders are available through any branch of Central Newsagency. In case of difficulty write to HOME COMPUTER ADVANCED COURSE BINDERS, Intermag, PO Box 57394, Springfield 2137.
Note – Binders and back numbers are obtainable subject to availability of stocks. Whilst every attempt is made to keep the price of the issues and binders constant, the publishers reserve the right to increase the stated prices at any time when circumstances dictate. Binders depicted in this publication are those produced for the UK market only and may not necessarily be identical to binders produced for sale outside the UK. Binders and issues may be subject to import duty and/or local taxes, which are not included in the above prices unless stated.

SUBJECT MATTERS

Computers have two distinct roles to play within the school system. As well as the study of computer science itself, we can consider the computer as a teaching aid — especially where it is used as an interactive textbook. Many educational publishers are now producing software specifically designed to cover examination syllabuses.

In 1980 the government of the United Kingdom launched a scheme designed to encourage computer literacy in both primary and secondary schools. Known as the Microcomputers in Education Project (MEP), it was set to run for six years with an overall budget of £21 million. It is perhaps unfair to divide this sum by 25,000 (the number of primary, middle and secondary schools in England alone), and thus assume even distribution of funds. The machine most strongly recommended was the BBC Microcomputer, which sells for £400. The second choice, Research Machines' 380Z, costs some five times as much. It is quite obvious that the budget for the project was hopelessly inadequate, and schools were forced back onto their own resources.

By 1983, midway through the scheme, the Minister for Information Technology was able to boast that all the secondary schools in the country (4,553 in England) had a computer, as did more than half the primary schools. However, much of the credit for this must go in fact to parents' associations, charitable trusts and not least the schoolchildren themselves, whose fund-raising efforts were considerable.

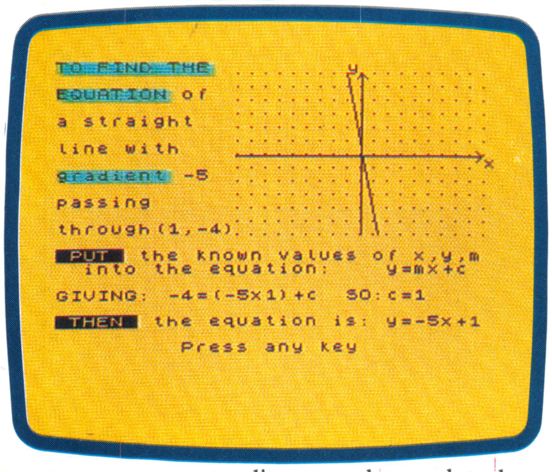
Rather than update the school curriculum and increase its relevance, the government initiative appears to have exacerbated existing problems. The need for expensive equipment has increased the frustration of teachers and pupils alike. There is now a keen awareness of the widening 'advantage gap' between children from comparatively wealthy backgrounds, who have more money to contribute to projects like this, and those from less well off families. Due to the small number of microcomputers available in any one school, the 'average' pupil is unlikely to have access to a machine for more than 15 minutes a week — which is hardly enough time to make a

Starting Right
Many home computer owners express their aspirations for their children's future in the purchase of educational software. For young children there is a wealth of material available to start them off on the learning process, and for the student sitting GCE and CSE examinations a greater number of very specific curriculum-based revision packages for a variety of microcomputers

ALAN ADLER

Revision Division
Software packages designed to help the student with revision for specific examinations are, quite properly, limited in scope. Each one attempts to cover a single aspect of the subject (mathematics, for example, or English literature) in considerable detail. Our first example deals only with mathematical equations of various sorts (linear, quadratic and simultaneous), and runs on the Spectrum, while the geometry package designed for the same hardware is wider reaching. The two remaining examples pose problems relating to Ohm's Law, and the design of amplifiers

'Equations And Inequalities', Rose Software



```
TO FIND THE
EQUATION of
a straight
line with
gradient -5
passing
through(1,-4)
PUT the known values of x,y,m
  into the equation:     y=mx+c
GIVING: -4=(-5x1)+c   SO:c=1
THEN the equation is: y=-5x+1
          Press any key
```

start on a computer literacy scheme, let alone explore the possible advantages of interactive educational software packages.

Given this inadequacy in the school system, it is perhaps less than surprising that a significant number of home computer owners cite the possibility of improving on their childrens' educational opportunities as one of the chief reasons for buying a machine.

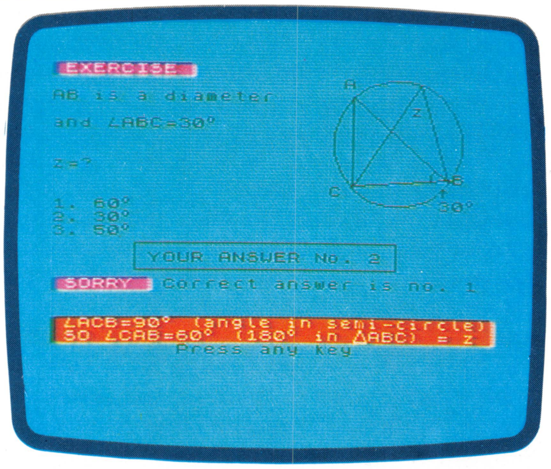
But leaving aside the problem of hardware, there is no doubt that educational programs play an increasingly important role as teacher aids in the classroom. Writing revision software, as it has come to be called, is a straightforward matter, requiring few of the tricks and devices employed in screen–based games programs, for instance. We have considered them as interactive textbooks because they have a great deal in common with their printed and bound counterparts. The identity of the author, for example, is an important selling point, just as it is with a conventional textbook, and the narrative pattern established in the book trade is also continued into the new medium.

The key difference, however, is the introduction of the concept of interaction. Traditionally, students are expected to rely on two sources of knowledge: the teacher and the textbook. The relationship with a teacher is interactive to some degree — albeit only to a small extent, given that classes are often 30-strong or more, and two hours of instruction per week allows each child four minutes of the teacher's undivided attention at the very best. It is not surprising that educationalists have explored the possibilities of more effective ways for children to interact with the material they are learning.

The first attempts at mechanised learning were the 'language laboratories' of the 1960s, which were applied to the teaching of imitative subjects such as foreign languages. Each student had access to a tape recorder and a pre-recorded text, and worked through that text at his or her own pace. The instructor had access to each student's individual audio channel, and could monitor or intervene as necessary. But the objective was to structure the course of instruction in such a way as to make that intervention as infrequent as possible.

Computer-based instruction takes this to its logical conclusion, and removes the necessity for an instructor to intervene at any time. In many ways the revision package must be an expert system: that is, it must be totally self-contained, include no inaccuracies and, furthermore, be designed in such a way as to lead the user through the material easily and naturally. In this instance

'Geometry', Rose Software



```
EXERCISE
AB is a diameter
and ∠ABC=30°

Z=?

1. 60°
2. 30°
3. 50°
          YOUR ANSWER No. 2
SORRY  Correct answer is no. 1

∠ACB=90°  (angle in semi-circle)
SO ∠CAB=60°  (180° in ΔABC) = Z
          Press any key
```

the computer system should be as easy to use as possible. It must assume very little in the way of computer literacy on the user's part, and certainly must not require any of the skills of a computer programmer or operator in order to make it usable.

When we come to consider the content of the software itself, then we must first differentiate between the sciences (where we are dealing with known and quantifiable facts) and the arts (where much of the analysis is judgemental and subjective). In the case of computer-based revision packages, there is a further distinction between the arts and the sciences. The latter will tend to be highly illustrated — even where we must rely on microcomputers that possess relatively low resolution graphics capabilities — whereas the arts subjects will be forced to rely very heavily on text-handling routines.

In addition to the very curriculum-specific tutorial and revision packages, a considerable body of material has been produced in the field of primary education — especially in simple arithmetic, reading and spelling.

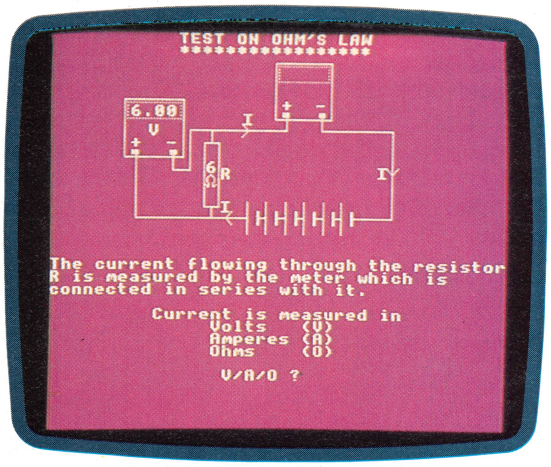
So far we have concerned ourselves with computer software that is either didactic or inquisitorial in nature; either seeking to present facts in such a way as to make them easy to memorise, or asking multiple choice questions based on that information. However, there is a third type of software, more generally used in the classroom than by the student at home: experimental simulation software. In this case the programming techniques involved are much more complex, as the program is required to replicate mathematically the interaction of physical forces.

The application of simulation methods to the classroom is proving as popular with teachers of science as it is with their counterparts in industry, for the simple reason that it allows a much greater degree of experimentation within the class — especially in areas where either the cost, or the dangers inherent in the use of reactive chemicals, would otherwise render it impossible.

Let's now consider the three major age groupings for which educational packages are produced, in order to compare the similarities and differences of the software.

EIGHT AND UNDER

Most software packages designed for the younger child are concerned with the development of basic skills in shape and pattern recognition. Many use simple arithmetic and spelling as the objects to be recognised or matched, thus reinforcing the familiarity of these symbols. Many programs that fall within this category make use of games-like 'scripts' in an attempt to hold the child's attention more securely. Some assume the presence of an adult or older child.


'D.C.', SciCAL Software

```
TEST ON OHM'S LAW
*****************

The current flowing through the resistor
R is measured by the meter which is
connected in series with it.

       Current is measured in
            Volts    (V)
            Amperes  (A)
            Ohms     (O)

            V/A/O ?
```

Particularly popular amongst this age group are programs that teach the child how to tell the time, practise counting, add and subtract (one interesting method involves an animated balance, which tips one way or the other depending on the load in the pans), construct short sentences and spell common words — often using the Hangman game as a stratagem.

NINE TO FOURTEEN

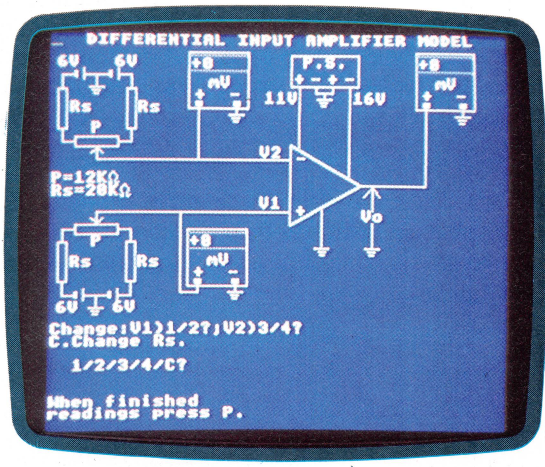
Within this age group the emphasis changes slightly, away from learning through play towards a rather more disciplined approach, and thus replicates the child's experiences in the classroom. Not surprisingly, given that software of this type is expected to keep the child motivated to use it, a considerable amount of effort has been devoted to incentives and rewards. One method that has proved popular and reasonably successful presents the child with a screen-based game after he or she has completed a section of the learning program within the time allowed and with a given measure of success.

Arithmetic and spelling and the use of language are still the most popular subjects for children in this age group. But in addition there are a variety of historical and geographical packages available, some that teach basic theory of music and a number of simple simulations.

FIFTEEN AND ABOVE

It is at this stage, where General Certificate of Education and Certificate of Secondary Education examinations loom, that the expert system type of educational software comes into its own. Basic skills reinforcement packages are available, of course, but more producers concentrate their efforts on particular subject areas. These are geared not just to a particular examination paper, but, in the case of GCE revision software, to the syllabus set by a particular Examination Board. At this level individual programs will typically offer a resumé of expected background knowledge, give the theory and methodology of the topic under consideration, and then provide a lengthy multiple choice test based on the subject matter. In subjects such as English literature they will also offer an analysis of style and content, just as a teacher would.

By this stage the student is expected to have reached a certain level of self-motivation, and little or no effort is made to keep him or her interested in the task at hand by artificial means.


'Ampli.', SciCAL Software

```
DIFFERENTIAL INPUT AMPLIFIER MODEL

P=12KΩ
Rs=20KΩ

Change:V1)1/2?;V2)3/4?
C.Change Rs.
  1/2/3/4/C?

When finished
readings press P.
```

IAN McKINNELL

This is not to say that the subject matter is generally presented simply at face value. On the contrary, an inventive approach to static and animated graphics and the use of synthesised sound is universally encouraged. There is a wide variety of software packages available for this age group, and as many are very subject-specific it is wise to consult a professional dealing in educational software in order to decide on their relative value.

In a future instalment of The Home Computer Advanced Course, we will look in more detail at the range of educational software packages available for the most popular home computers.

FUNCTIONS AND CONTROL STRUCTURE

In the first part of our appraisal of the Sinclair version of the BASIC language, we dealt with Sinclair's idiosyncratic approach to variable names and string-handling. Here we conclude our look at the dialect by considering the VAL, GOSUB and GOTO functions, and the control structures WHILE...WEND and REPEAT...UNTIL.

You may already have noticed that some functions in Sinclair BASIC do not require brackets around their arguments, unlike their counterparts in other BASICS, so that LEN(X$) can be written as either LEN X$ or LEN(X$). You should, however, use brackets if the meaning of an expression is doubtful or ambiguous.

The function CODE is the Sinclair equivalent of ASC(), and behaves in exactly the same way. The Sinclair character set, however, is standard ASCII only for values 32 to 122. So, for example, where PRINT CHR$(7) in most BASICS causes a 'beep' of some kind to be sounded, in Sinclair BASIC it causes an error message.

The function VAL is standard BASIC, but in Sinclair BASIC a statement such as VAL("a45") would cause the program to crash because the argument of the function is non-numeric. In most other BASICS this would simply return the value zero. If this particular quirk is likely to be a problem you may have to write a subroutine to replace the VAL function, or you may have to test the CODE value of the first character of the argument of VAL: if CODE A$(1) <48 or CODE A$(1) >57 then A$ is non-numeric and should not be the argument of the function VAL.

Sinclair VAL has the unusual power, however, of evaluating numeric expressions, so that:

```
LET A$="6*12":PRINT VAL A$
```

will result in 72, the value of the expression '6*12'. In most BASICS, VAL is not as powerful as this, and would return the value 6 in this case.

This ability to evaluate expressions can be put to use in many ways. A simple example is this block graph drawing program:

```
100 DIM S$(31)
200 LET S$="*******************************"
300 INPUT "ENTER A FUNCTION OF X";F$
400 PRINT "Y = ";F$:PRINT
500 FOR X=1 TO 10
600 PRINT S$( TO INT(VAL F$))
700 NEXT X
800 PRINT"=============================="
900 PRINT "000000000111111111122222222223"
950 PRINT "123456789012345678901234567890"
```

When you run this program, you may type in any algebraic expression with variable X (2*X+3/X, for example), and you will see a crude block graph of that function on the screen. In this program, the y-axis is horizontal, the x-axis vertical, and graphics are pixel-size. It would not require much more code to establish the range of values of x and y immediately after the function of x was input, and then make scaling adjustments, print axes and create high resolution graphics. The result would be a very impressive graphics package; the fragment above, however, is simply designed to make clear the usefulness of VAL's power to accept expressions as its argument.

Just like VAL, in this respect, are GOSUB and GOTO. They too evaluate expressions in situations where most BASIC dialects would require that their arguments be valid line numbers. This has several advantages. You can give your subroutines names, for example, define variables with the same names and appropriate values, and then GOSUB to the subroutines by name. Here is an example of this ability:

```
100 LET INIT=1000
200 LET OUTPUT=2000
300 LET CALCULATE=3000
400 GOSUB INIT
500 GOSUB CALCULATE
600 GOSUB OUTPUT
700 STOP
1000 REM*******INIT S/R********
.............
2000 REM*****OUTPUT S/R********
.............
3000 REM****CALCULATE S/R******
```

which almost makes your program self-documenting. If you replace GOSUB INIT by GOSUB (VAR1+N*VAR2), or any valid numerical expression, the expression will be evaluated and the result treated as a line number. Furthermore, in Sinclair BASIC, if the argument of GOTO or GOSUB is a line number that does not exist, then control passes to the next valid line number. If you write, for example, GOTO 17 and line 17 does not exist but line 18 does, then control will pass to line 18 and no error will result — as it does in most BASICS.

The ability of Sinclair BASIC to handle these 'computed jumps' makes up for the lack of the ON...GOTO and ON...GOSUB structures. In another BASIC you might write:

```
2360 ON D GOSUB 100,200,300,400,500
```

meaning that if the value of D is 1 then the program will jump to line 100 (GOSUB 100), if D=2 then it will jump to line 200 (GOSUB 200), and so on. In

Sinclair BASIC you would be able to write:

```
2360 GOSUB (100*D)
```

to produce exactly the same effect. In both versions, however, you must consider what will happen if the value of D is non-integral, less than one, or greater than five.

Spectrum BASIC does not support the control structures REPEAT...UNTIL, and WHILE...WEND. These can, however, be simulated in several ways. The REPEAT structure is a loop starting at the word REPEAT, and finishing at the word UNTIL, which is followed by a conditional statement. If this condition is true then the loop terminates and control passes to the statement after UNTIL. On the Spectrum, this control structure can be simulated in this way:

```
100 DATA "A","B","C","D","E","*","F","G","*"
200 FOR L=1 TO 1
300 READ X$
400 PRINT X$
500 IF X$<>"*" THEN LET L=0
550 NEXT L
600 PRINT"END OF DATA"
```

The problem with the REPEAT...UNTIL structure is that, because the exit condition is tested at the end of the loop, the loop will always be executed at least once. If this is undesirable, then the WHILE...WEND structure should be used.

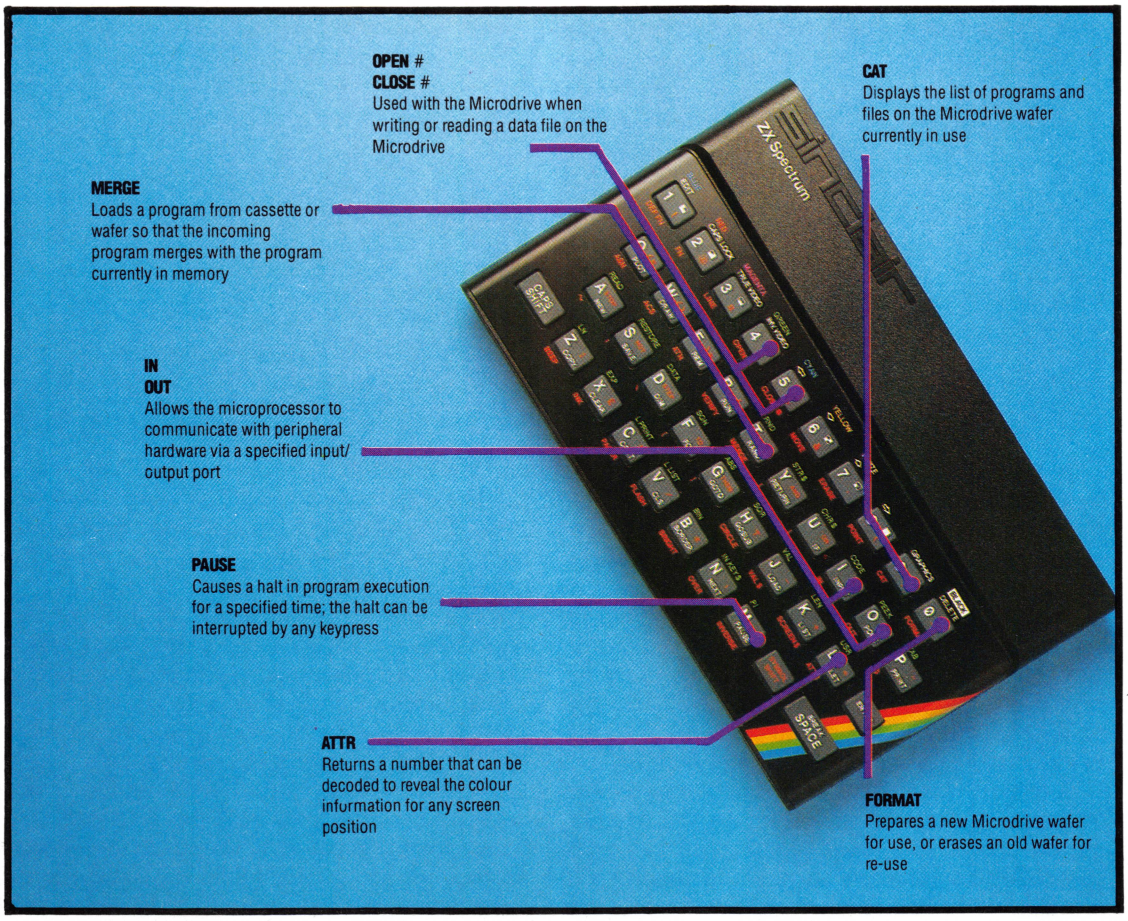
This structure is a loop starting at the WHILE statement (which is followed by a condition) and ending at the W(HILE)END statement. As long as the condition is untrue, the program statements between WHILE and WEND will be executed repeatedly. When the condition is true, control passes from the WHILE statement to the statement after WEND (the statements in between are skipped over). The program given, therefore, would read:

```
100 DATA "A","B","C","D","E","*","F","G","*"
200 WHILE X$<>"*" DO
300 READ X$
400 PRINT X$
500 WEND
600 PRINT"END OF DATA"
```

This could be replaced on the Sinclair machines by an IF...THEN...GOTO structure, but another alternative is:

```
50 LET X$=""
100 DATA "A","B","C","D","E","*","F","G","*"
200 LET TEST=(X$<>"*")
250 FOR L=1 TO TEST
300 READ X$
400 PRINT X$
500 LET L=(X$="*")
550 NEXT L
600 PRINT"END OF DATA"
```

We have dealt with the major variations in Sinclair BASIC. There are other smaller differences, some of which are bugs rather than genuine changes, but with care and the handbook you should be able to cope.



OPEN #
CLOSE #
Used with the Microdrive when writing or reading a data file on the Microdrive
CAT
Displays the list of programs and files on the Microdrive wafer currently in use
MERGE
Loads a program from cassette or wafer so that the incoming program merges with the program currently in memory
IN
OUT
Allows the microprocessor to communicate with peripheral hardware via a specified input/output port
PAUSE
Causes a halt in program execution for a specified time; the halt can be interrupted by any keypress
ATTR
Returns a number that can be decoded to reveal the colour information for any screen position
FORMAT
Prepares a new Microdrive wafer for use, or erases an old wafer for re-use

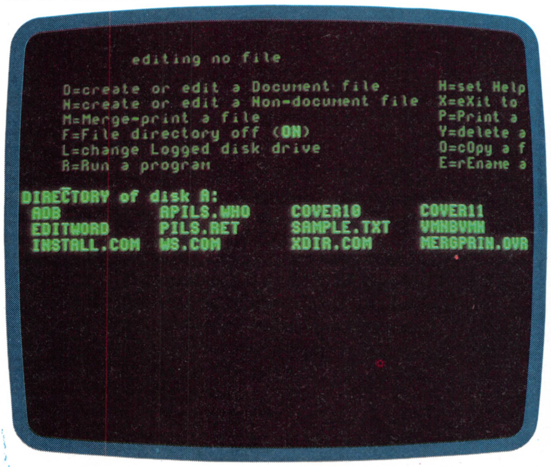
IAN McKINNELL

SCREEN WRITERS

Word processing is the largest single applications area in professional microcomputing — and a surprisingly large number of home computer users also have acquired such software. In this series, we will be looking at a variety of word processing software packages, starting with the world-wide best seller: WordStar.

In its simplest form, a word processor is no more than a sophisticated typewriter. The monitor screen replaces the paper, and the Delete key acts as an eraser. For the business user, the most desirable feature of a word processing system is the ability to save documents — on disk or perhaps cassette tape — and then allow those documents (which might be single lines, sentences, paragraphs or whole pages) to be incorporated into newly generated text. This is especially useful in the production of specifications, contracts and other documents that use the same passage over and over again.

The No-File Menu



```
        editing no file

   D=create or edit a Document file        H=set Help
   N=create or edit a Non-document file    X=eXit to
   M=Merge-print a file                    P=Print a
   F=File directory off (ON)               Y=delete a
   L=change Logged disk drive              O=cOpy a f
   R=Run a program                         E=rEname a

DIRECTORY of disk A:
 ADB          APILS.WHO    COVER10      COVER11
 EDITWORD     PILS.RET     SAMPLE.TXT   VMNBVMN
 INSTALL.COM  WS.COM       XDIR.COM     MERGPRIN.OVR
```

The No-File Menu
To begin, WordStar presents a menu showing the choice of its basic functions. The user could create or edit either a document file using all the standard formatting parameters, such as wordwrap, justification, etc., or a non-document file, or copy, rename or delete a file. In short, this section of the program deals with housekeeping routines. These pictures were taken from the Osborne-1, which displays only 52 of the available 80 columns of text, hence the right hand cut-off

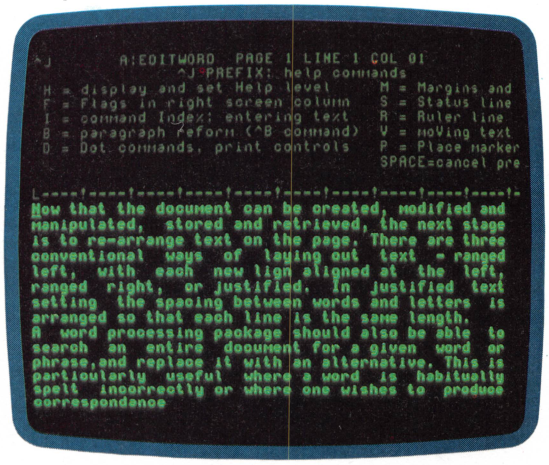
The Help Menu
Accessed by the Control [J] code, this menu offers the user a series of fairly comprehensive on-screen explanations of the way in which certain sections of the package operate. It is colloquially known as the Help menu. Following the J control code with one of the subsidiary characters — H, B, F, D etc. — delivers a fuller explanation of those sections of WordStar

Now that the document can be created, modified and manipulated, stored and retrieved, the next aspect to look at is how to re-arrange text on the page. There are three conventional ways of laying out text: ranged left (where each line is aligned at the left), ranged right, or justified. In justified printed text the spacing between words and letters is arranged so that each line is the same length. Most printers that are available for microcomputers, however, vary only the spacing between words when producing justified text.

A word processing package should also be able to search an entire document for a given word or phrase, and replace it with an alternative. This is particularly useful where a word is habitually spelt incorrectly, or where one wishes to produce correspondence personalised with a name. In technical terms, WordStar is a menu-driven word processor and text editor with embedded transparent formatting commands, running in a CP/M environment, and requiring the use of the Control key to enable the machine to differentiate between a command and a character. This means the operator can choose to have a menu of options constantly displayed on the screen, in this case in a reserved section 10 lines deep at the top. Commands that affect the way text appears — either at input time or when it is printed — are displayed within the main body on the screen but not reproduced by the printer; and file handling and 'housekeeping' responsibility is assumed by the operating system.

Once loaded, WordStar presents the user with the 'no-file menu', which allows the selection of the required function. The user can create or edit a file, either as a document or as a 'non-document' (used, for example, in Assembly language programs that are to be compiled, or for text that is to be used by another software

The Help Menu



```
^J        A:EDITWORD  PAGE 1 LINE 1 COL 01
                 ^J PREFIX: help commands
 H = display and set Help level        M = Margins and
 F = Flags in right screen column      S = Status line
 I = command Index: entering text      R = Ruler line
 B = paragraph reform (^B command)     V = moVing text
 D = Dot commands, print controls      P = Place marker
                                       SPACE=cancel pre

L----!----!----!----!----!----!----!----!----!----!-
Now that the document can be created, modified and
manipulated,  stored and retrieved, the next stage
is to re-arrange text on the page. There are three
conventional  ways  of  laying out  text  - ranged
left,  with  each  new lign aligned at  the  left,
ranged  right,  or justified.  In  justified  text
setting  the spacing between words and letters  is
arranged so that each line is the same length.
A  word processing package should also be able  to
search  an  entire  document for a given  word  or
phrase,and replace it with an alternative. This is
particularly  useful  where a word  is  habitually
spelt  incorrectly or where one wishes to  produce
correspondance
```

package); merge files together (with or without printing them); change the disk drive in use from A to B, or vice versa; delete, copy or rename a file; exit to the operating system; or set the level of help required.

Each option is selected by a single key depression (no need for the Control key), leading the user to a smaller menu specific to the function in use. The last of these options (to set the help level) allows the user to select the extent to which the menu of available commands will be displayed on the screen. The lower the level of help required, the fewer the lines reserved on the screen for the menu, and the more available for text editing. Most of the initial key commands are mnemonic — as much as they can be with a single character. For instance, D creates a Document file and N creates a Non-document file.

The first request from the system is for a name for the file that you are about to create. Having named your file, the system switches into the working menu, which consists of prompts for such things as cursor movement, insertion and deletion, and reminders of the single-character control codes used to enter other branches of the main menu (the formatting menu, for example). WordStar does not assume that the machine on which it will be run (the 'object computer') will have the cursor control keys.

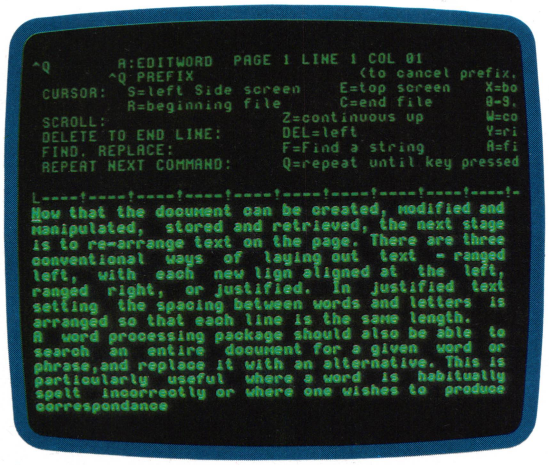
Single-character space movement is defined by the Control key in conjunction with the E, S, D and X keys (machines that have cursor control keys normally have WordStar installed so that the functions are duplicated). WordStar is rather powerful in its cursor control, accommodating cursor movement by character, word, line or paragraph in either direction. Much the same principle is applied to deletion. Single characters can be deleted either at the cursor position or immediately to the right of it; and whole words to the right of the cursor position or entire lines (irrespective of cursor position) can be deleted. The screen can also be scrolled using the control functions, without affecting the positioning of the cursor. This is very useful if you wish to refer back in the text.

Lástly, the user can decide whether to insert a character at the cursor position, or replace the character with the one entered from the keyboard. This function is 'toggled' — that is, one depression of the appropriate key (V in this case) changes the status from insertion to replacement, and there it stays until the V key is pressed again (in concert with the Control key), when the status will revert.

The package automatically looks after pagination and wordwrap at line ends. Pagination defines the number of lines per page and wordwrap is the automatic taking-over of words to the next line.

The functions described are all that one normally needs for creating and editing documents, but WordStar has other features as well. It can centre lines on the page; enable multiple striking to produce emboldened characters; give superscript and subscript characters; overprint the previous line; or underline and strike out individual characters. All these functions are available at text entry time or while editing.

The Quick Command Menu



```
^Q        A:EDITWORD  PAGE 1 LINE 1 COL 01
         ^Q PREFIX                        (to cancel prefix,
 CURSOR:  S=left Side screen     E=top screen      X=bo
          R=beginning file       C=end file        0-9,
 SCROLL:                      Z=continuous up      W=co
 DELETE TO END LINE:          DEL=left             Y=ri
 FIND. REPLACE:               F=Find a string      A=fi
 REPEAT NEXT COMMAND:         Q=repeat until key pressed

L----!----!----!----!----!----!----!----!----!----!-
Now that the document can be created, modified and
manipulated,  stored and retrieved, the next stage
is to re-arrange text on the page. There are three
conventional  ways  of  laying out  text  - ranged
left,  with  each  new lign aligned at  the  left,
ranged  right,  or justified.  In  justified  text
setting  the spacing between words and letters  is
arranged so that each line is the same length.
A  word processing package should also be able  to
search  an  entire  document for a given  word  or
phrase,and replace it with an alternative. This is
particularly  useful  where a word  is  habitually
spelt  incorrectly or where one wishes to  produce
correspondance
```

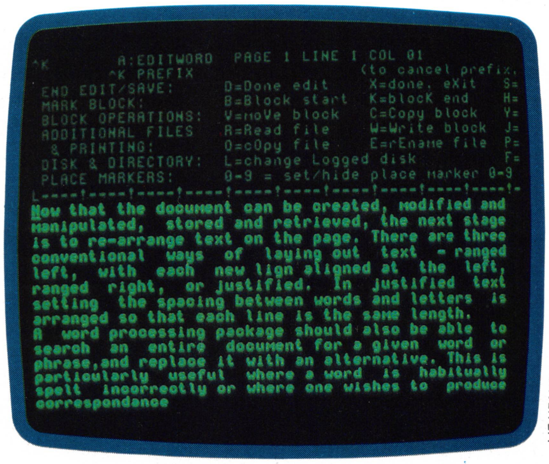
The 'find and replace' function is accessible via the Q menu, which also contains the extra cursor positioning and delete functions. The user is requested to enter the string of characters making up the word or phrase that is to be searched for. He can nominate a particular occurence of it, a specific number of occurences, or indeed all instances, and decide whether the string located is simply to be pointed out by the cursor, or to be replaced, and if so whether that replacement should take place without further instruction.

Marked blocks of text can similarly be moved from one place in the document to another by means of various functions of the K menu, which otherwise deals with file handling. WordStar's three other subsidiary menus, the J, O and P menus, deal respectively with assistance to the user in creating or editing a document; formatting and arrangement of documents; and the way they are to appear when printed.

Overall, WordStar is a comprehensive, professional microcomputer software package. There are very few functions that it cannot happily perform, and it has been defined in such a way as to be accessible to a wide variety of machines. It might seem at first sight to be complicated and idiosyncratic, but thanks to its comprehensive menus, even the beginner can make use of it.

A significant number of the microcomputers that will support WordStar have programmable function keys as standard. The Osborne-1, for instance, on which our examples have been displayed, offers the user 10 of these keys, and reserves 256 bytes for their contents. These functions — accessed by means of the Control key and a numeric key pressed simultaneously — can contain a string of control characters, or a commonly used word or phrase, and enable that character string to be incorporated by just two key depressions.

The Block Menu



```
^K        A:EDITWORD  PAGE 1 LINE 1 COL 01
         ^K PREFIX                        (to cancel prefix,
 END EDIT/SAVE:         D=Done edit      X=done, eXit     S=
 MARK BLOCK:            B=Block start    K=blocK end      H=
 BLOCK OPERATIONS:      V=moVe block     C=Copy block     Y=
 ADDITIONAL FILES       R=Read file      W=Write block    J=
  & PRINTING:           O=cOpy file      E=rEname file    P=
 DISK & DIRECTORY:      L=change Logged disk              F=
 PLACE MARKERS:         0-9 = set/hide place marker 0-9
L----!----!----!----!----!----!----!----!----!----!-
Now that the document can be created, modified and
manipulated,  stored and retrieved, the next stage
is to re-arrange text on the page. There are three
conventional  ways  of  laying out  text  - ranged
left,  with  each  new lign aligned at  the  left,
ranged  right,  or justified.  In  justified  text
setting  the spacing between words and letters  is
arranged so that each line is the same length.
A  word processing package should also be able  to
search  an  entire  document for a given  word  or
phrase,and replace it with an alternative. This is
particularly  useful  where a word  is  habitually
spelt  incorrectly or where one wishes to  produce
correspondance
```

LIZ HEANEY

The Quick Command Menu
WordStar's secondary menus are entered bý means of a single character control code, which is to some degree acronymous. The Q menu, for example, is the short way of saying the Quick menu. The Q menu provides the user with a resumé of the cursor control keys, as well as the package's find and replace functions

The Block Menu
Some of the single character acronyms are rather obscure — K, for example, stands for blocK. It controls the saving and utilisation of files, the movement of blocks of text and some file access structures. Note that it replicates some of the package's standard housekeeping routines as well

A

A-D CONVERTOR

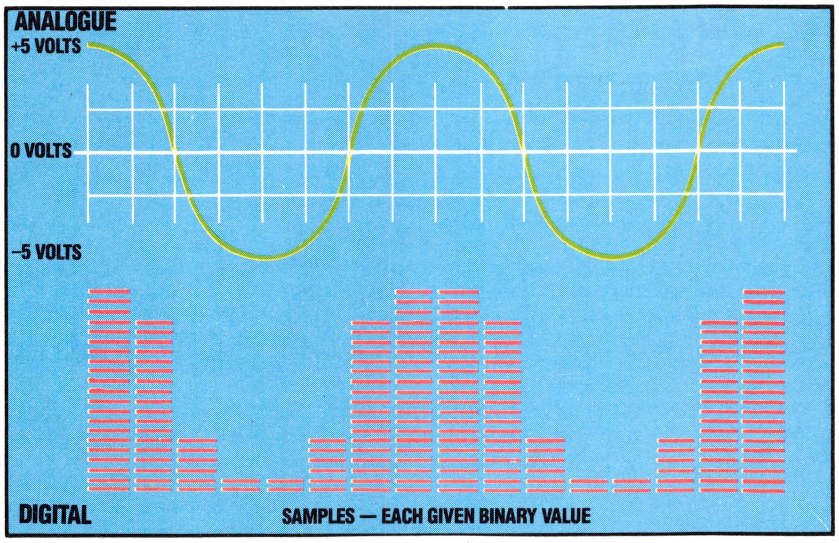
An analogue-to-digital (usually abbreviated to A-D) convertor is a hardware device that converts electrical signals in analogue form into their digital equivalent. Digital-to-analogue (D-A) convertors perform the opposite process, but the circuitry involved is quite different.

To understand the purpose of such devices, we need to appreciate the precise difference between the terms 'analogue' and 'digital'. A digital signal (by signal we mean a varying voltage) has two properties: first, it is 'discrete', which means that it can take on only one of a number of predefined values. Inside a home computer, for example, the digital signals can assume only two values — 0 volts or 5 volts — corresponding to logical zero or one. Secondly, a digital signal is almost always 'encoded': any value is represented as a collection or series of discrete values. You should already be familiar with the idea of the binary system, where a collection of eight bits can represent a value in the range 0 to 255.

An analogue signal, by contrast, is 'continuous' rather than discrete. Within a range defined by the maximum and minimum limits, the signal can take on any one of an infinite number of possible values. Secondly, as the name suggests, an analogue signal is always an analogy of a measurable quantity. For example, a thermocouple will output a voltage in proportion to the temperature it measures, and a microphone produces a fluctuating voltage in proportion to the level of sound.

There are computers that process analogue signals directly (called analogue computers) but these are expensive and are limited mostly to engineering applications. It is far easier to perform mathematical and logical processes on digital signals, which is why almost all computers are now digital.

A-D and D-A convertors are therefore used to interface digital computers with analogue devices such as those mentioned above. Some home computers, such as the BBC Model B, have an A-D convertor built in, which increases the machine's appeal to school science departments. It is possible to buy A-D (and, less commonly, D-A) convertors for many home computers.



LIZ DIXON

ADDER

An adder is a logic circuit constructed from elementary logic gates (AND, OR and NOT) that will add two binary values together and produce an output.

The simplest adder is called a half-adder, which features two single-bit inputs (called A and B, say) and two outputs: sum and carry. The latter is needed because if both A and B take the binary value 1, then their sum will be 10 (i.e. the sum will be set to 0 and carry to 1).

A full adder is one that can accept a carry bit as an input in addition to A and B. This means that it is possible to chain any number of full adders together (connecting the carry output of one to the carry input of the one on the left) and thereby create a circuit that can add, for example, one eight-bit binary number to another, to create an eight-bit result (with the possibility of a ninth bit or final carry).

ADDRESS

When programming in BASIC a memory address is seldom needed, but in machine code it is vital. Every microprocessor (CPU) can talk to a finite number of memory locations (each location is a byte) and this is referred to as the addressing range — 65,536 on a standard eight-bit home computer. Each byte will have a unique number, label or address by which the CPU can refer to it and thereby read or alter the contents or the data that it contains. To read the contents of, for example, byte number 47,339 the CPU must put the binary equivalent of this address onto the group of 16 wires called its *address bus.* The selected byte puts a copy of its eight bits onto a separate collection of lines called the data bus, from which the CPU can read them into one of its internal registers.

ADSR

Even the earliest computers designed specifically for home use had some form of sound output, and the better models allowed you to alter the volume of the noise or music produced, under software control. ADSR is a facility introduced on the more sophisticated home computers that allows the programmer to control not only the tone or frequency of the note (or notes, if it has more than one voice) produced, but also the timbre or type of sound.

It is properly called Attack Decay Sustain Release envelope control, meaning that you have control over the way the volume of a note changes as it is played. The envelope is the curve or shape of the graph of volume against time. Values are given to each of the four phases of a note: attack (as it rises in volume), decay (as it falls off), sustain (the volume at which the note is sustained), and its release at a specified rate.

IN THE PICTURE

For the occasional game or business letter the quality of the image on the screen may not be particularly critical. If, however, a computer is used seriously for business or education, then the advantages offered by a monitor over a television set become significant — and the problem becomes one of the right system for your needs.

One of the most important peripherals associated with computers is the screen. In many cases this is the television set, or a cheap black-and-white set bought specially, and though the quality obtained is acceptable, it isn't as good as it could be.

This is because the signal, in passing from the screen memory inside the computer to the screen, has to undergo several stages of encoding and decoding. No matter how good the circuitry involved, it will inevitably be less than perfect, with the result that the final image is often smeary and hard to read, and frequently has a most unpleasant shimmering effect, which is known as 'dot-crawl'.

The secret of really high quality displays is therefore to eliminate these signal distortions, and since they are produced by the modulating and demodulating circuitry this can be achieved simply by leaving these processes out of the system entirely.

In effect that is what a monitor does. It is a television tube without the television decoders, a simplified system that is able to generate sharper, brighter and generally more stable images.

Since the television decoders are absent, the monitor will not perform if it is connected to the television output socket on your computer. You need a Video Out socket. It may not be labelled as such, but the important detail is that it must be output that does not pass through the modulator, and to check this you should consult your computer manual.

The process of generating an image on a television screen is largely a matter of making sure that everything happens at the right moment. The problem lies in the fact that the sweep of the beam across the face of the tube is generated inside the monitor, and is thus inaccessible to the driving device in the computer itself.

Turned on but left unconnected to an input, a monitor will scan the beam over the whole screen 50 times a second, producing a perfectly even illuminated field. Turning this into a picture involves switching this beam on and off in exactly the same place every fiftieth of a second, and any instability in this process will result in a most unnerving shimmering, which makes the screen at best exhausting to look at and at worst unusable.

The whole process depends upon 'synchronisation pulses', which together with the brightness signal are produced directly by the computer and output to the monitor.

There are two types of 'sync-pulse': one for each line of the picture, and one for each complete picture. At the end of each complete cycle the monitor is sent a short pulse telling it that the frame is now complete and that the electron beam (and thus the dot that it produces) must be returned to the top left-hand corner of the frame to repeat the cycle.

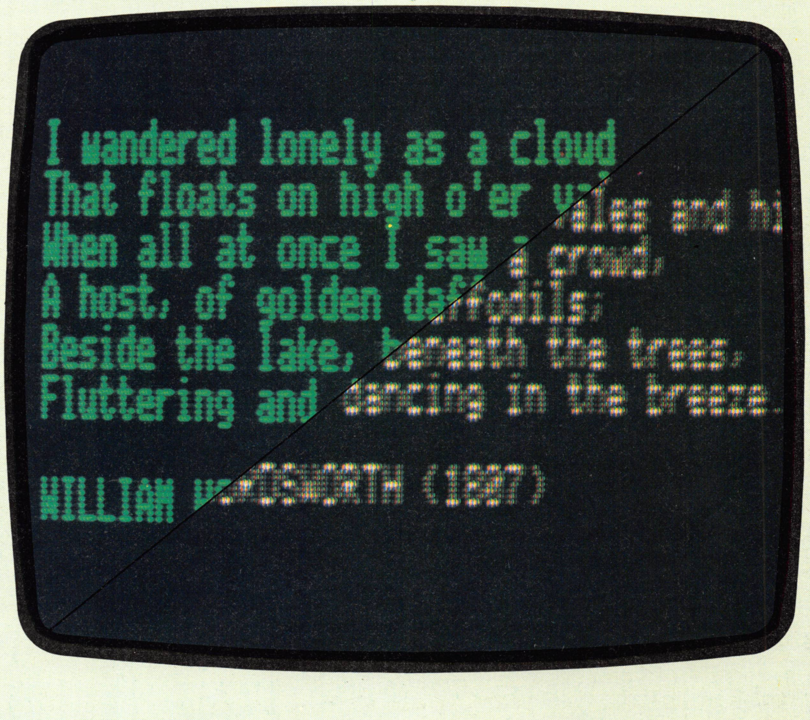
A similar event occurs at the end of each line. The monitor receives a pulse telling it that a line is complete, and that the electron beam must be returned to the left-hand side of the screen to start on the next line.

There are several kinds of monitor, but they fall into two main categories, colour and monochrome, subdivided into the different types of signal that they accept.

Monochrome monitors are quite simple, and

A Picture Of Dedication

These pictures show the quite dramatic difference between the image quality attainable with a dedicated monochrome monitor (in this case Apple's Monitor III) and a high quality domestic television set when it is used to display the output from a word processing package


```
I wandered lonely as a cloud
That floats on high o'er vales and hi
When all at once I saw a crowd,
A host, of golden daffodils;
Beside the lake, beneath the trees,
Fluttering and dancing in the breeze.

WILLIAM WORDSWORTH (1807)
```
IAN McKINNELL

the only important type of signal is called 'composite', meaning that the various timing pulses are combined with the brightness level into one signal, which the monitor then sorts out to produce the picture.

COLOUR MONITORS

A similar scheme is used in some colour monitors, but because of the greater complexity of the signal these are more closely related to television sets and operate on much the same systems. The main types are PAL, SECAM and NTSC, the names of the colour systems used in television broadcasting in different parts of the world. They represent methods of encoding the three 'additive' colours (red, green and blue) and the two synchronisation pulses.

Alternatively, the various signals can be sent to the monitor separately. Though there are subtle variations in the schemes used, they are generically known as RGB, after the colour guns (red, green, blue). The simplest is TTL (transistor-transistor logic), in which the colours have only two states, on or off, and so in various combinations can produce the familiar eight colours seen on viewdata systems such as Prestel and Ceefax.

More colours can be produced if each additive colour can be given an intensity, and though these intensities are usually given in discrete (digital) steps, the result is known as an 'analogue' or

Bandwidth

Bandwidth is an important consideration in your choice of a monitor. It is a measure of how small a dot the circuitry can produce. To an extent, the smaller the dot the better, but the decision as to what bandwidth is best for a given machine is based on other considerations as well.

Your priority is to work out what the absolute minimum bandwith could be for your machine. This is calculated by multiplying the maximum possible number of character lines on the screen by the number of characters on each line, giving the total number of characters. This figure is multiplied in turn by the width of the character matrix in dots, then by the depth of the matrix.

The result is the maximum number of dots on the screen, which is generally somewhere between 10,000 and 1.5 million. On an ordinary 80 × 24 screen with a 7 × 9 matrix the figure is 110,960. All these dots are illuminated by the 'raster' on the screen at a rate of 50 times a second (60 in the US and many other countries), which means that the number of dots on the screen must be multiplied by this figure. The result will generally lie between 500,000 and 75,000,000, and this is the number of times the controlling circuitry must be able to turn the beam on and off per second.

This is a frequency, so is expressed in hertz (Hz), and because the figures are almost all in millions the units are Megahertz — Meg for short.

Monitors are readily available with bandwidths of 5, 7, 10, 12 and 15MHz, and 20MHz is possible, though the price rises sharply as the figure increases. Monitors operating at 100MHz can be built but they cost many thousands of pounds.

In general, the higher the bandwidth of your monitor, the clearer its image, although too much resolution can be as unwelcome as too little, particularly in colour graphics.

If the bandwidth of the monitor is too low for the computer, the letters will be smudgy and hard to read, as though the signal were being muddied up. If the bandwidth is too high, individual dots will become too clearly separated, breaking the image up.

Monochrome Monitors
The simplest and least expensive of all monitors are th monochrome models. For £100 or less, these units offer the user a choice of phosphor colours and reasonably high definition. They are ideally suited to word processing and other business applications, bu are often used to display simple graphics, as well

Three-Colour Monitors
Microvitec's CUB is an excellent example of the less expensive colour monitors available to the home computer user. While its resolution is not of the highest, it is quite suitable for a variety of applications including most graphics

Versatility From Japan
With its ability to decide on the standard of the received signal, JVC's TM90PSN is particularly valuable to those who wish to use their monitor in a variety of different applications. In addition to its use as a computer output device, it can also service a video recorder or video disc player

Domestic Televisions
The picture quality of domestic televisions has improved rapidly with advances in cathode ray tube construction technology and electron gun alignment. Sony, for example, has shown its awareness of the dual use to which its sets are put by installing the antenna connector on the front of the set

CHRIS STEVENS

'linear' monitor.

The price of monitors ranges upwards from about £70 for a nine-inch black-and-white set. A 12-inch phosphor-green screen monitor costs about £110 to £150. Colour monitors are more expensive, owing to the higher cost of the tube. The 'standard' monitors such as the Microvitec and Kaga ranges cost from £200 to about £350, depending largely on the size of the screen and the bandwidth. Most colour monitors are available in both TTL and analogue versions, with little difference in the cost.

The TM90PSN is an interesting and flexible monitor produced by the JVC company. Costing about £320, it is smaller than most, with a 10-inch screen, and though the bandwidth is not particularly high (which limits the maximum resolution), it is able to accept almost any type of input, ranging from ordinary monochrome composite, through TTL and analogue RGB to any one of the four composite colour signals, PAL, SECAM, NTSC 3.58 and NTSC 4.43. It has a selector mechanism inside which checks the incoming signal and automatically switches the monitor to the appropriate mode.

This kind of monitor will become more common, and it can serve as a general-purpose output for computers, video recorders, video disc players and other machines.

Persistence And Colour

An important factor in choosing a monochrome monitor is the type of phosphor used. This is the powdery substance that coats the inside of the screen itself and has the property of glowing when struck by an electron beam, thus producing the picture.

The main considerations are the colour and the 'persistence'. The first is self-explanatory, but persistence is less well understood. It is a measure of the length of time for which the phosphor will continue to glow after the beam has passed on. Its precise value is seldom stated and it is usually simplified to 'long' and 'short' persistence.

The phosphors used in television sets and virtually all colour monitors have a short persistence, as do the majority of monochrome monitors, but in many applications it is easier for the viewer if the image continues to glow for a fraction of a second, since this reduces screen flicker, a major cause of eyestrain.

The best-known example of long-persistence phosphor is seen on radar screens, on which the radial sweep of the beam leaves a long glowing trail, giving the screen a kind of 'memory' without the need for complicated electronics.

However, if the system employs a light pen, long persistence is a liability — in fact, it would make the light pen unusable, since its operation is dependent on the image fading fast enough for the pen to be locatable by the computer. If the phosphor continues to glow, the computer will see light no matter where the pen is pointed and regardless of whether the beam is actually scanning that point or not. Hence the computer will not be able to calculate the position of the light pen using its knowledge of which position on the screen is being scanned at a particular moment.

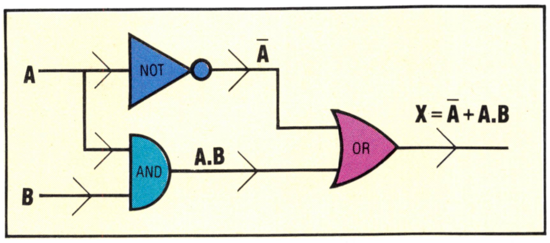
A range of types and colours of phosphor are employed, depending on the use to which the monitor will be put, and on the preference of the purchaser. Almost any colour can be produced. White phosphor with short persistence is cheap and readily available, but one of the shades of green is almost as common and is much easier on the eye, as is amber. Blue is seen on many mainframe terminals such as those used by travel agents and on airline desks, while red is used in radar rooms and other places in which night vision must not be impaired.

THE BUILDING BLOCKS OF ADDITION

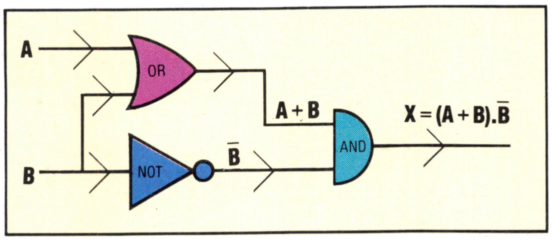
In the first instalment of this course, we looked at the three kinds of logical building blocks (AND, OR, and NOT) and saw how these may be combined together to produce simple logic circuits. Now we begin to investigate the way in which these logic circuits can be used to perform the function of addition.

The system of using algebraic notation to describe logical relationships is known as Boolean algebra, and is named after the mathematician George Boole (1815–1864). Boolean algebra is of great use in computer circuit design because it allows mathematical simplification of logic circuits. This means that fewer logic gates are required to perform a given function, which in turn increases the speed of operation of the machine.

We have already met the Boolean notation for the output from the three basic logic gates: AND (A.B), OR (A+B), and NOT ($\overline{A}$). More complex circuits can be represented by using these expressions. For example, the Boolean expression $X = \overline{A} + A.B$ represents this circuit:



It is important to notice that the order in which the operations AND and OR are carried out is significant. The simple rule is that AND has priority over OR (as well as over NOT). If it is wished to reverse the order of priority then brackets have to be used, as in this example: $X = (A+B).\overline{B}$. For this expression the two inputs are ORed together before being ANDed with the negative of B. Here is the circuit diagram:



When drawing a logic circuit from its Boolean expression it is often best to start at the output and work backwards to the inputs. This method produces better circuit diagram layouts.

INCLUSIVE AND EXCLUSIVE OR

There are two possible meanings of the word OR in everyday English. The first meaning is the one that we have already met, namely:

One OR the other OR both.

The second meaning has important consequences for logic circuit design:

One OR the other but not both.

For example, to qualify for a competition for two-wheeled vehicle owners you could own a motorcycle OR a bicycle (or you may own both). This is a case of the *inclusive* use of OR. On the other hand, you may be tall OR you may be short (but you can't be both). In this case the use of the word OR *excludes* the possibility of both statements being true.

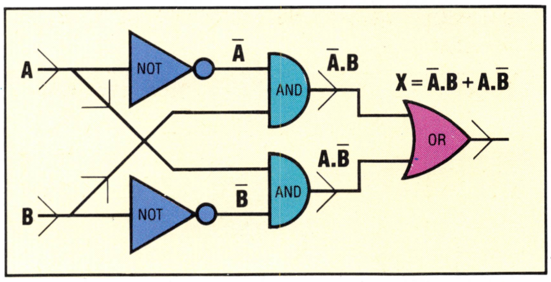
In logic circuits this *exclusive* OR (or XOR) operation is a useful tool and can be built up from the AND, OR and NOT set of gates. The truth table for XOR is:

INPUTS		OUTPUTS
A	B	$A \veebar B$
0	0	0
0	1	1
1	0	1
1	1	0

As can be seen from the second and third rows of the truth table, a value of one can be produced as output if:

NOT(A) is ANDed with B
OR
A is ANDed with NOT(B).

This can be written as a Boolean expression in this way: $X = \overline{A}.B + A.\overline{B}$. A possible circuit to produce the Exclusive-OR operation from the above Boolean expression would be:



This would produce a five gate circuit. We shall see later how it is possible to simplify this circuit to one with only four gates using Boolean algebra.

LOGIC GATES AND ARITHMETIC

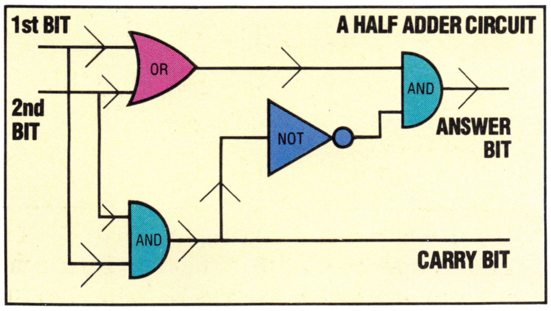
Although most home computers can perform the complete range of arithmetic functions, only addition is directly carried out by logic circuits. All other functions, such as subtraction, multiplication and division, are carried out using a combination of hardware 'adder' circuits and software to control the movement of bit patterns. Before we can look at the circuits required to perform binary addition we must look at the process itself. Let us consider the following binary addition:

	8s	4s	2s	1s
	0	1	0	1
	0	1	1	1
	1	1	0	0
CARRIES	1	1	1	

If we take one column of the sum in isolation — say the twos column — we can list the various inputs and outputs to and from that column. The *inputs* are: the two bits to be added, and the carry bit from the previous column. The *outputs* are: the bit put into the twos column in the answer, and the bit to be carried into the next column. A device that would accept these inputs and produce the correct outputs is known as an *adder*. This device is rather complex so let us start with a slightly simplified version, known as a *half adder*. A half adder circuit ignores the fact that there may be a carry from a previous column. This reduces the problem to a circuit with two inputs and two outputs. We can now devise a truth table for a half adder circuit, which looks like this:

INPUTS		OUTPUTS	
1st BIT	2nd BIT	CARRY BIT	ANSWER BIT
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

It is easy to see that the carry bit will be one if the first bit AND the second bit are both one. The answer bit is formed by ORing the inputs together, except in the case when the carry bit is one. We may describe the answer bit output as being one if: 'the first bit is one OR the second bit is one, and NOT if the carry bit is one'. The following circuit will produce the required outputs:



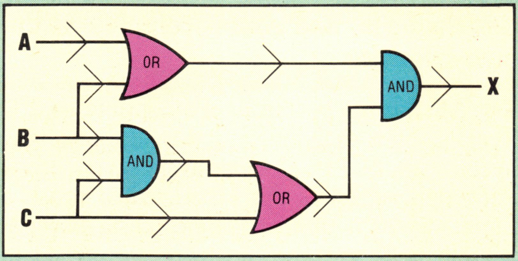
EXERCISE 2

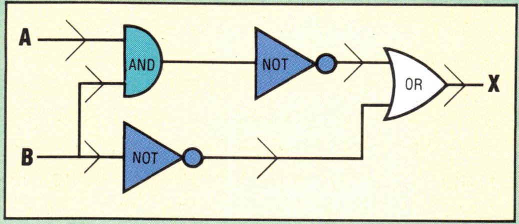
1) Draw logic circuits for the following Boolean expressions:

a) $X = (A + B).C$
b) $X = A.B + (A + C)$
c) $X = \overline{A}.B + (A + B)$
d) $X = \overline{A.B}.(A + B)$

2) Write down the Boolean expression using A and B as inputs for the carry bit and the answer bit of a half adder circuit.

3) Write down the Boolean expressions for these circuits:





Answers to Exercise 1 On Page 9

1)

PASSED DRIVING TEST	ACCOMPANIED BY QUALIFIED DRIVER	ALLOWED TO DRIVE
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

2)

CASSETTE PLAYER AVAILABLE	DISK DRIVE AVAILABLE	WRITTEN FOR DIFFERENT COMPUTER	PROGRAM WILL LOAD
FALSE	FALSE	FALSE	FALSE
FALSE	FALSE	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE
FALSE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE
TRUE	FALSE	TRUE	FALSE
TRUE	TRUE	FALSE	TRUE
TRUE	TRUE	TRUE	FALSE

3)

A	B	P	Q	C
0	0	1	0	0
0	1	1	1	1
1	0	0	1	0
1	1	0	1	0

LOOP THE LOOP

The Sinclair concept of low-cost, high-power computing was achieved with the immensely popular Spectrum home computer, but this micro lacked a large-capacity, fast-access storage device until Sinclair introduced the Microdrive. It costs less than £80 and comes complete with its own interface.

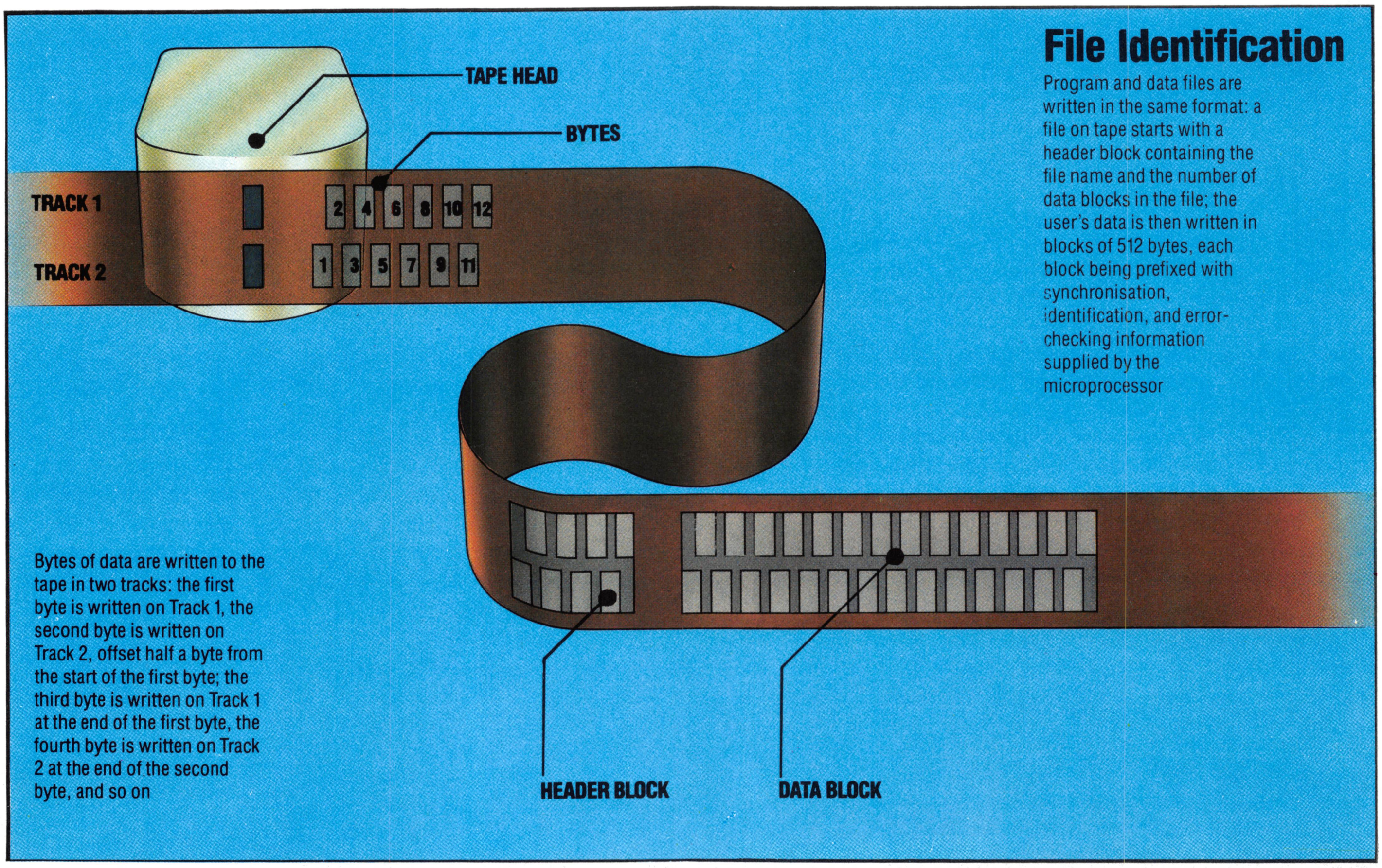
The Microdrive system uses a continuous 200in loop of 2mm magnetic video tape that revolves inside a small cartridge once every seven seconds. Although accessed and instructed in much the same way as a standard floppy disk drive, it is actually a 'floppy tape' or 'stringy floppy'. Data is recorded digitally, as opposed to the audio tape method, which records tones. The video tape is formatted into two tracks in which data bits are recorded sequentially in an overlapping zig-zag pattern. This system, which requires two read/write heads, enables data to be stored at double the density and speed of a single-track system. The tracks are further formatted into blocks of 512 bytes. Each block contains a description of the data contained within it and is preceded by a 'header' consisting of 27 bytes of identifying data.

A block and its associated header are known as a sector. The 200in tape loop can store almost 200 sectors, giving a data storage density of approximately 500 bytes per inch. A file identified with a file name is stored in a single sector if it is less than 512 bytes in length. If it overlaps into another sector, or several, without filling the final sector, the unused sector space is lost until the file is erased. So although each Microdrive can theoretically store 100 Kbytes of data, its capacity is in reality between 85 and 90 Kbytes. Average access time to find and load a program into Spectrum memory is between 10 and 15 seconds.

THE ZX INTERFACE

The Microdrive is connected to the Spectrum via a ZX Interface 1, which is attached to the micro's peripheral edge connector. Up to eight Microdrives can be connected one to another in a 'daisy chain' arrangement. In addition to providing a connection for Microdrives the Interface 1 serves as a standard RS232 interface, a local area network interface and a ZX Printer connector. It also contains instructions within ROM that extend Sinclair BASIC to include suitable data handling commands for the Microdrive and other interfaces.

The ZX Microdrive and the Interface 1 are powerful additions to the Spectrum system. The facilities offered closely simulate the operation of



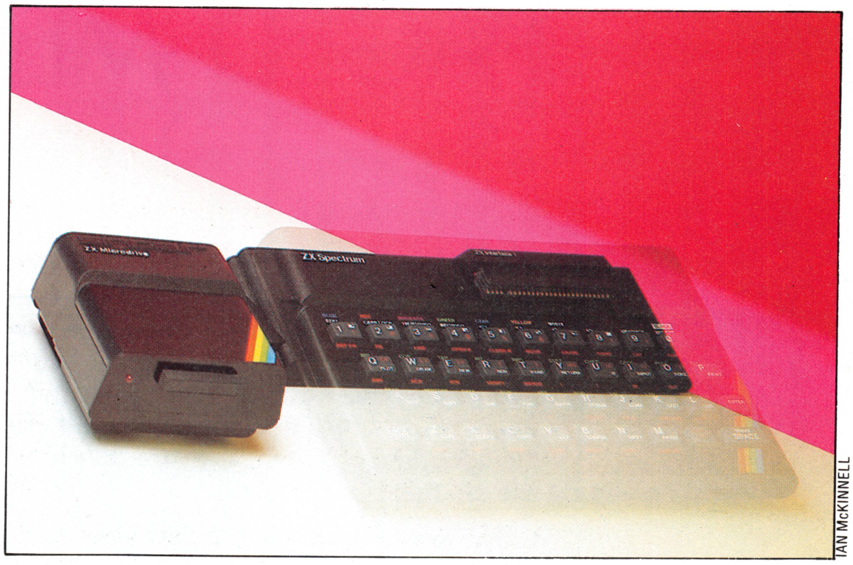
File Identification

Program and data files are written in the same format: a file on tape starts with a header block containing the file name and the number of data blocks in the file; the user's data is then written in blocks of 512 bytes, each block being prefixed with synchronisation, identification, and error-checking information supplied by the microprocessor

Bytes of data are written to the tape in two tracks: the first byte is written on Track 1, the second byte is written on Track 2, offset half a byte from the start of the first byte; the third byte is written on Track 1 at the end of the first byte, the fourth byte is written on Track 2 at the end of the second byte, and so on

ALUN JONES

a conventional floppy disk system. However, limitations are imposed by the dubious long-term reliability of the tape cartridges and the lack of a true random access file facility. This affects the viability of the Microdrive as a storage system for serious business applications, for most commercial software demands the location, storage and retrieval of small amounts of data many times during use. Moreover, little software recorded on Microdrive cartridge is available at present, although this is likely to change dramatically in the near future. The advent of the Sinclair QL business computer, supplied with two built-in Microdrives, may well bring about an extension of the capabilities of the system. If Microdrives prove their reliability they will be of enormous value to the Spectrum owner who requires simply the facility to store large amounts of data and retrieve it quickly.


IAN McKINNELL

Stringy Floppy
Combining the cheapness and simplicity of tape storage with the speed of the disk, the Microdrive has become known as a stringy floppy. It connects to the Spectrum via the Interface 1. Phono sockets for cassette access and an RS232 port are also features of the Interface 1

Microdrive Commands

Commands that are relevant to the Microdrive are:

```
FORMAT CAT     SAVE*   VERIFY* LOAD*
MERGE* ERASE   OPEN#   PRINT#  INPUT#
INKEY$# CLOSE# MOVE
```

In each case M selects Microdrive, N is the number of the Microdrive to be accessed (1–8) and S is the stream number allocated (4–15).

Format
Formats the tape data layout, names the cartridge and erases all data previously stored. It can be constructed as follows:

```
FORMAT "M";N;"NAME" or
FORMAT M$;N;C$
```

where NAME is the name selected for cartridge (1–10 characters), M$ (either M or m) is as for M, and C$ (1–10 characters) is as for NAME.

Cat
This command loads to screen or specified stream a catalogue of all the files contained on cartridge in a specified drive. This is achieved with:

```
CAT N or
CAT#S;N
```

The catalogue contains the cartridge name, up to 50 filenames and free space on cartridge in Kbytes.

Save*
This creates program files that can be programs, named strings or data, and is constructed in one of the following ways:

```
1. SAVE*"M";N;"FILENAME"
2. SAVE*"M";N;"FILENAME"SCREEN$
3. SAVE*"M";N;"FILENAME"DATA A()
4. SAVE*"M";N;"FILENAME"LINE X
```

creating the following:

1. A file
2. A file consisting of SCREEN$
3. A file consisting of data A()
4. A file that will RUN from line X on LOADing.

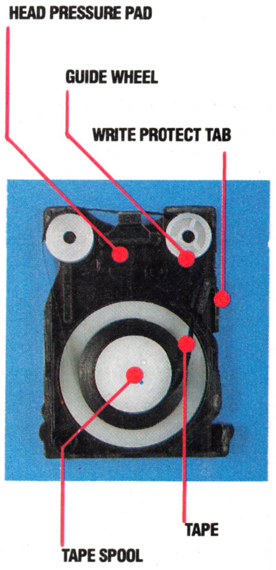
Verify*, Merge* And Erase*
These commands are constructed in the same manner as 1. SAVE* above. VERIFY* compares the file "FILENAME" with the current contents of user memory and generates an error message if they are different. MERGE* merges "FILENAME" with the current contents of user memory, and ERASE* erases "FILENAME".

Load*
LOAD* can be constructed in the same manner as 1. SAVE* and 2. SAVE*. When executed, LOAD* copies the contents of the specified file into user memory.

Open#, Print#, INPUT#, Inkey$#, Close# And Move
These commands are concerned with the handling of data files. Data files are stored sequentially but can be manipulated to mimic some of the properties of random access files by allowing a data file to be read in its entirety and extracting the required data after loading. In the same manner, the data file, then in user memory, can be modified and re-stored. Data file handling is organised by OPENing and CLOSEing streams to establish data channels that can be accessed by reference to the specified stream number. For example:

```
OPEN#S;"M";N;"FILENAME"
```

connects stream S to file "FILENAME" on Microdrive N. "FILENAME" can then be written to by using PRINT#S and read from by using INPUT#S or INKEY$#S. MOVE can be used to transfer data files within a cartridge, from one Microdrive to another, or to any device that can be accessed via a stream number. When stream channels are no longer required, they should be disconnected with CLOSE#S.



Microdrive Wafer
The tape in the wafer is a 2mm video tape, used for its strength, and high storage density. The tape forms an 8-metre loop and is coated so that it slips easily from between the metal tape spool and the wound tape. The read/write head presses the tape against the head pressure pad. When in place, a write-protect tab enables data to be read from or written to the wafer; if the tab is removed, the wafer can be read only, thus protecting the information stored

PAGED MEMORY

As the next step towards understanding the fundamentals of machine code programming, we must examine the way computers organise and manage their memory. Here, we look at the constraints imposed on both memory pagination and the operation of the CPU by the machine's use of the binary system.

In the first instalment of the Machine Code course we gave an analogy of the way in which a computer stores information in the form of electric current. We used the example of a factory where each worker had an individual switch pattern that lit up four light bulbs in the manager's office, thus identifying who was at work. This showed how information (i.e. the name of the person who is working) could be represented by using a flow of electricity.

In our example, we found that by using four switches and bulbs we could represent the numbers from 0 to 15. In other words, there were only 16 possible patterns. However, if we had used eight switches and bulbs instead, then we could have made 256 unique patterns (2×2×2×2×2×2×2×2=256) and, therefore, have been able to count from 0 to 255.

In your home computer, memory is arranged in individual banks of eight switches, and each of those eight-switch banks is called a *byte*. In general, the CPU handles information one byte at a time, which means, in effect, that it can only add, compare, and store numbers between 0 and 255. This might seem to limit its arithmetical capabilities, but that isn't the case. If you think about doing a sum like 63951 + 48770 = ? then you will see that you actually manipulate the individual digits one at a time. Similarly, the CPU can perform arithmetic on large numbers using one byte at a time.

Because it has eight switches, a byte is a place where an eight-digit binary number can be stored. Each of these binary digit positions is called a *bit* — the smallest possible unit of information. A bit in a byte is either ON or OFF, a binary digit is either 1 or 0.

It's often important to talk about individual bits in a byte, so the convention is to number the bits 0 to 7 from right to left in the byte. If a byte contains the binary number 00000001, then we say that bit0 is 1, or that bit0 is ON, or that bit0 is SET; all the other bits are 0, or OFF, or CLEAR. Thus in the binary number 01001000: bit3 is SET, bit6 is SET, bit4 is OFF, bit7 is 0, bit0 is CLEAR, and so on. In a byte, bit0 is also called the Least Significant Bit (LSB), and bit7 the Most Significant Bit (MSB).

Computer memory, then, can be conceived as a long strip of squared paper, eight squares wide, and thousands of squares long: each row of eight squares is a byte, each square is a bit in a byte. Memory is useless if you can't locate items in it, so each of the bytes has an identifying label called its *address*; the address isn't written anywhere on the paper (or in the byte), it's simply the number of the byte in memory, counting from the start of memory. The first byte, therefore, has the address 0, the next byte has address 1, the next has address 2, and so on. If you want to write something in byte43, then you start at the bottom of the memory (at byte0) and count through the bytes until you reach byte43.

When you get there nothing will identify that byte as byte43 except for its position — you've counted forward from byte0, you've reached 43, so this must be byte43. The bytes of memory are actually minuscule banks of eight-transistor devices (one device per bit, eight devices per byte) etched into the chips inside your machine, and they are identical in everything except their physical position.

However, there is one drawback to this method. This system of memory addressing would be fine if there were only a few hundred bytes. The CPU can count from 0 to 100 in fractions of a millisecond; but computers have thousands of bytes, and counting from 0 to 20000 must take some appreciable time, even for a microprocessor. The way a computer overcomes this problem is by dividing memory into *pages*, just as books are.

If we continue to think of computer memory as a strip of squared paper thousands of squares long and eight squares wide, we can imagine cutting that strip on the boundary of every 100 bytes (i.e. cut across the boundary between byte99 and byte100, cut across the boundary of byte199 and byte200, byte 299 and byte 300, and so on). Each of the strips of paper between the cuts is now a page of 100 bytes. Page0 starts at byte0 and continues to byte99; page1 starts at byte100 and continues to byte199, page2 is byte200 to byte299, etc. Now to find any byte, say byte3518, we don't have to count 3518 bytes from the start of memory because we can see from the address that this byte must be on page 35. Therefore, we need only count 35 pages from the bottom of memory, and then count the bytes from the bottom of that page until we reach byte18 on the page, which must be byte3518. Try it with a strip of squared paper if you haven't followed this.

This system of paged memory is convenient because we can look at the address of any byte, and split it into two parts — the digits from the hundred column leftwards are the page number of the byte, and the digits from the tens column to the right are the number of bytes counted from the bottom of that page. In the example above we have actually split the address 3518 into two numbers: page number 35, and byte number 18 on that page. We call 18 an *offset*, or *page offset*, because it is the number by which you must offset (or increase) the address of the bottom byte on the page in order to reach the byte in question.

The computer, however, doesn't count in decimal, as we do — it counts in binary. The paging system depends upon being able to find the page and the offset by simply inspecting the address of the byte. The decimal address 99 is represented by 01100011 in binary, and 100 decimal is 01100100 binary; decimal 199 is 11000111, and decimal 200 is 11001000 binary. We can see, from these examples, that there's no simple way of looking at the binary numbers and telling page from page, as we can so easily do with the decimal equivalents. The reason for this is the choice of page size.

We chose 100 as the page size precisely because it's a meaningful number in the decimal system (it's a power of 10). If we are to count in binary, however, then we must choose a page size to suit *that* system. The page size used by our computers is 256, so that page0 starts with byte0 and continues to byte255; page1 starts with byte256 and continues to byte511, and so on. To see why this is convenient we must write these addresses in binary:

```
Page0: byte00000000  — byte11111111
Page1: byte100000000 — byte111111111
```

As you can see, we can count in binary from 0 to 255 in an eight-bit number; the next number — 256 — requires nine bits, and with nine bits we can count up to 511. The next number — 512 — requires ten bits, and with ten bits we can count up to 1023; and so on. We now see that if the page size is 256 and we count in binary, then the offset is the rightmost eight bits, and the page number is given by the bits from bit8 leftwards.

This may be puzzling since we have already stated that the CPU can handle only single bytes, and a byte contains only eight bits. Therefore, you may ask, what good is it to talk about nine- and 10-bit numbers? The answer is that all addresses in memory are treated as two-byte numbers, and the CPU deals with them one byte at a time. If we rewrite the page boundaries as two-byte numbers this system becomes more clear:

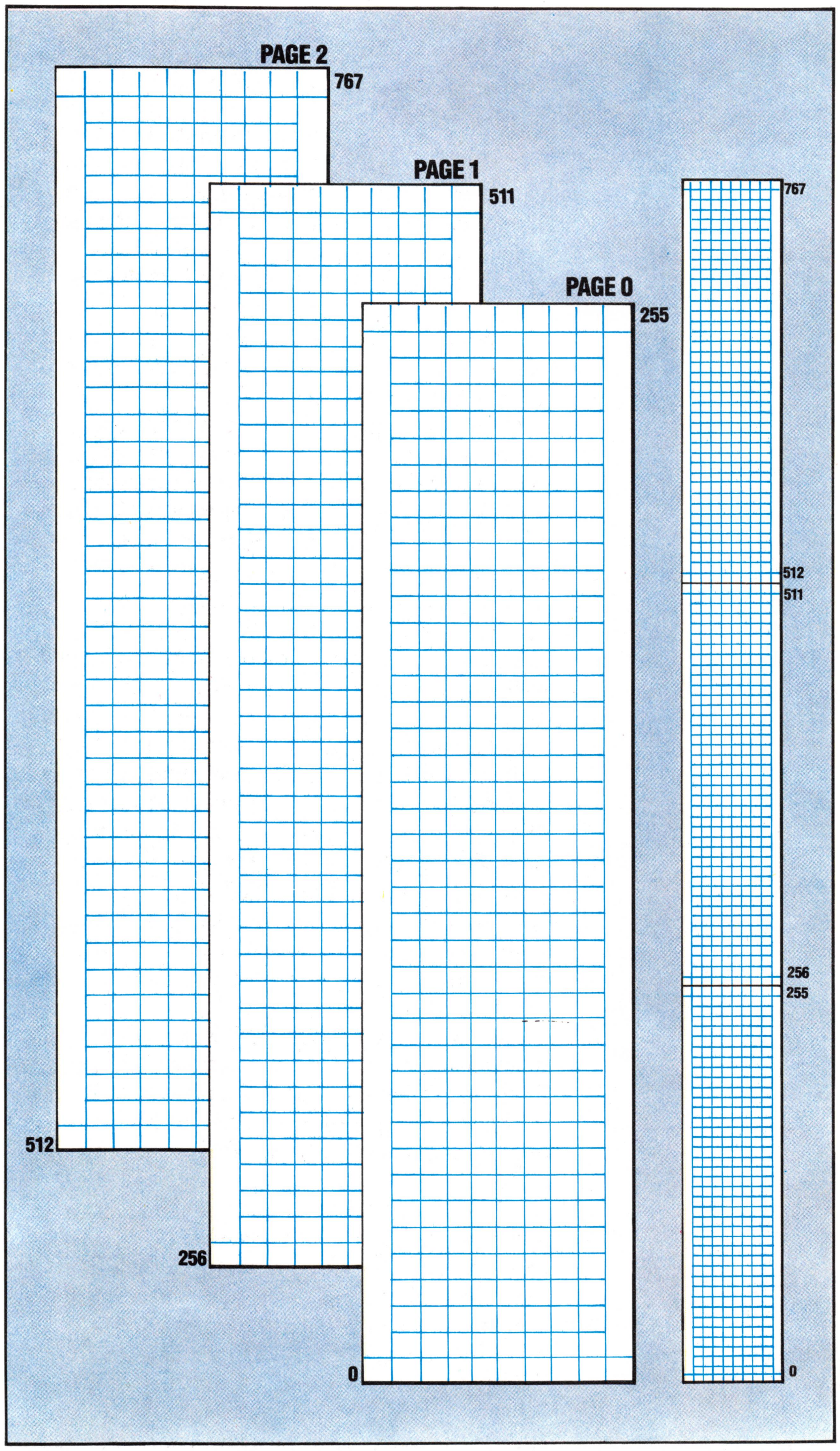
```
Page  0 starts at 00000000 00000000
        ends   at 00000000 11111111
Page  1 starts at 00000001 00000000
        ends   at 00000001 11111111
Page 10 starts at 00000010 00000000
        ends   at 00000010 11111111
Page 11 starts at 00000011 00000000
        ends   at 00000011 11111111
```

and so on.

Now we can see that when the CPU fetches information from, or puts information into, a byte in memory, that byte will be identified by a two-byte address. The first, or leftmost, byte of the two gives the page number, while the second, or rightmost, byte gives the offset.

On page 38, we provide programs that convert from decimal into binary, as well as hexadecimal numbers. The latter are used extensively in machine code, and will be fully discussed later in the course.

Paged Addressing
Paged addressing divides memory into imaginary blocks or pages of 256 bytes. All addresses are then expressed as two-byte numbers: one byte gives the page number, the other gives the offset from the start of that page



Number Cruncher

The three programs presented here, for the BBC Microcomputer, Spectrum and Commodore, accept decimal numbers, and deliver in return their binary equivalent

Commodore 64

```
10 REM********COMMODORE**********
40 S$="=
         ":X$="0123456789ABCDEF"
50 REM   S$ CONTAINS 9 SPACES
60 PRINT CHR$(147)      :REM CLEAR SCREEN
70 PRINT "
S"         TO DISPLAY DECIMAL NUMBER
80 PRINT "
TS"        AND THEIR BINARY EQUIVALEN
90 PRINT:PRINT "      *******ENTER 0 TO QUI
T********":PRINT
100 FOR K=1 TO 1
110 FOR L=1 TO 1
120 INPUT"TYPE ANY POSITIVE WHOLE NUMBER
 ";A$
130 NU=VAL(A$)
140 IF NU=0 THEN PRINT "PROGRAM EXIT":ST
OP
150 IF INT(NU)<>ABS(NU) THEN L=0
160 IF NU>65535 THEN PRINT NU;" IS TOO B
IG":L=0
170 NEXT L
200 NM=NU:H$="":GOSUB 2000
210 PRINT NU;TAB(5);N$;
220 IF RIGHT$(A$,1)="+" THEN GOSUB 4000
230 PRINT H$:PRINT:PRINT
240 K=0:NEXT K
250 END
300 END
1000 REM******BINARY BYTE S/R*******
1010 B$=""
1020 FOR D=8 TO 1 STEP-1
1030 N1=INT(N/2)
1040 R=N-2*N1
1050 B$=MID$(STR$(R),2)+B$
1060 N=N1
1070 NEXT D
1080 RETURN
2000 REM****BINARY CONVERSION S/R***
2010 IF NM<256 THEN N=NM:GOSUB 1000:N$=S
$+B$:RETURN
2020 HI=INT(NM/256):LO=NM-256*HI
2030 N=HI:GOSUB 1000:N$="=  "+B$
2040 N=LO:GOSUB 1000:N$=N$+"  "+B$
2050 RETURN
3000 REM****HEX BYTE S/R*********
3010 HB=INT(N/16):LB=N-HB*16
3020 B$=MID$(X$,HB+1,1)+MID$(X$,LB+1,1)
3030 RETURN
4000 REM****HEX CONVERSION S/R******
4010 IF NM<256 THEN N=NM:GOSUB 3000:H$="
=    "+B$:RETURN
4020 HI=INT(NM/256):LO=NM-256*HI
4030 N=HI:GOSUB 3000:H$="=  "+B$
4040 N=LO:GOSUB 3000:H$=H$+"  "+B$
4050 RETURN
```

BBC Micro

Copy the Commodore list with the following changes:

```
60 CLS:@%=5
210 PRINT TAB(0);NU;TAB(5);N$;
1050 B$=STR$(R)+B$
```

This program does not use the BBC's number representation facilities for the sake of compatibility of format with the other machines: you may be able to rewrite it in a shorter form.

Spectrum

```
10 REM*******SPECTRUM*********
40 LET S$="=            ":LET X$="0
123456789ABCDEF"
50 REM   S$ CONTAINS 9 SPACES
60 CLS
70 PRINT "  TO DISPLAY DECIMAL N
UMBERS"
80 PRINT " AND THEIR BINARY EQUI
VALENTS"
90 PRINT:PRINT " *******ENTER 0
TO QUIT*********":PRINT
100 FOR K=1 TO 1
110 FOR L=1 TO 1
120 INPUT"TYPE ANY POSITIVE WHOL
E NUMBER ";A$
130 LET NU=VAL(A$)
140 IF NU=0 THEN PRINT "PROGRAM
EXIT":STOP
150 IF INT(NU)<>ABS(NU) THEN LET
 L=0
160 IF NU>65535 THEN PRINT NU;"
IS TOO BIG":LET L=0
170 NEXT L
200 LET NM=NU:LET H$="":GOSUB 20
00
210 PRINT NU;TAB(5);N$;
220 IF A$(LEN A$)="+" THEN GOSUB
 4000
230 PRINT H$:PRINT:PRINT
240 LET K=0:NEXT K
300 END
1000 REM**BINARY BYTE S/R**
1010 LET B$=""
1020 FOR D=8 TO 1 STEP-1
1030 LET N1=INT(N/2)
1040 LET R=N-2*N1
1050 LET B$=STR$(R)+B$
1060 LET N=N1
1070 NEXT D
1080 RETURN
2000 REM**BINARY CONVERS S/R**
2010 IF NM<256 THEN LET N=NM:GOS
UB 1000:LET N$=S$+B$:RETURN
2020 LET HI=INT(NM/256):LET LO=N
M-256*HI
2030 LET N=HI:GOSUB 1000:LET N$=
"=  "+B$
2040 LET N=LO:GOSUB 1000:LET N$=
N$+"  "+B$
2050 RETURN
3000 REM****HEX BYTE S/R******
3010 LET HB=INT(N/16):LET LB=N-H
B*16
3020 LET B$=X$(HB+1)+X$(LB+1)
3030 RETURN
4000 REM***HEX CONVERS S/R****
4010 IF NM<256 THEN LET N=NM:GOS
UB 3000:LET H$="=    "+B$:RETURN
4020 LET HI=INT(NM/256):LET LO=N
M-256*HI
4030 LET N=HI:GOSUB 3000:LET H$=
"=  "+B$
4040 LET N=LO:GOSUB 3000:LET H$=
H$+"  "+B$
4050 RETURN
```

If you input a number with a "+" on the end, for example: 6435+, then in addition to its decimal and binary representation, its hex representation will also be displayed.

STEP UP AND PLAY

Atari, like so many other home computer companies, started out as the brainchild of one individual. When Nolan Bushnell attached a microprocessor controller to his television set and invented the game Pong (which was, as its name suggests, roughly one half of a game of ping pong), he could have had no idea of the consequences.

Nolan Bushnell's simple method of placing control over what appeared on the screen in the hands of those watching it, was to transform the popular idea of leisure and capture the imaginations (not to mention the pocket money) of millions of youngsters.

Bushnell and his two partners, Ted Dabney and Larry Bryan, each put in a stake of £100 to launch Pong. The game made its first appearance in Sunnyvale, California in 1972 and soon showed signs of becoming a profit-generating sensation. Atari's domination of the home video game market dates from an astute decision made soon after, to buy the rights for Bushnell's invention.

Atari maintained its market lead for most of the 1970s, until public taste moved on from arcade game machines to home micros. Marketing games is like marketing records: you must spot the potential pop stars and promote them. It is apt, therefore, that Atari should be owned by the international conglomerate Warner Communications International, best known for its film and record industry interests. And although Atari's coin-operated arcade business brought in bumper profits in the late 1970s, takings dropped off by about 25 per cent in 1983 and involved the parent company in huge losses.

Taito's Space Invaders game, skilfully marketed by Atari, is the best known of all computer games. It became a social phenomenon and spawned a whole universe of intergalactic zapping games. Atari was at the centre of the arcade games boom of the late 1970s. The company's hits just kept coming: Asteroids, Battlezone, Centipede, Lunar Lander, Missile Command, and The Tempest.

But the arcade boom dropped off just as dramatically as it had begun. Customers turned to home computers because they offered two major advantages. You could play the same games on them as you could in the arcades, but in this case free of charge; and you had a very flexible computing machine, as well.

At first Atari responded to this change in market demand by converting its best arcade software into home computer games. These used solid-state cartridges that plugged into the back of a home computer unit, and either added to or supplanted the computer's own ROM. Although this proved to be an effective way of buying a computer game, as it did not demand that the player load the game program into memory from cassette or disk, the solid-state components meant that cartridges were very expensive. And because these cartridges were not reprogrammable (the programs were physically etched into the circuits), the company was often left with piles of electronic scrap from those games that proved to be unpopular.

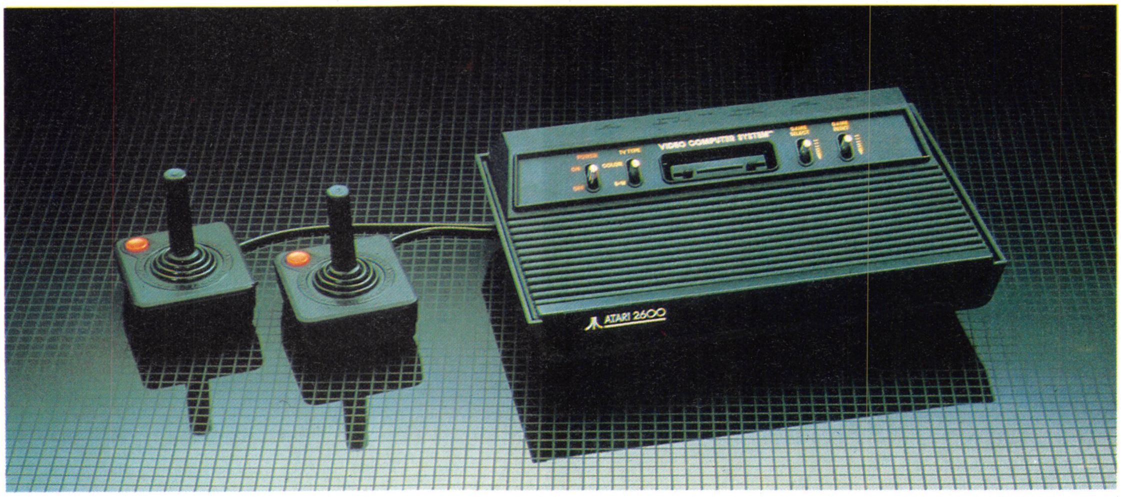
DECLINING FORTUNES

Atari's marketing judgement soon showed signs of weakening. The company based the sales projections for some of its games cartridges on those of the phenomenally successful PacMan game, and eventually paid the price for this miscalculation. An inventory was made of the unsaleable cartridges, all priced between $8 and $25, and 14 truckloads of these were consigned to a large hole in the Nevada desert.

COURTESY OF ATARI

Nolan Bushnell
Atari's fortunes are based on the endeavours of one man — Nolan Bushnell. When Bushnell created Pong (the first computer game) in 1971, it is doubtful if he knew the nature of the Pandora's Box that he was opening

Atari also failed to capitalise on a unique product feature of computer games: computer code does not need to be converted into a physical entity to be effectively distributed. It can be transmitted over the telephone or by cable, or broadcast on radio or television. New techniques and products to permit these forms of transmission are becoming increasingly available. In 1983, for example, the Romox Corporation in the US unveiled a machine that they called the Romox Programming Terminal. This was a 15 Megabyte hard disk machine that could



Dedicated Games Console
For about £70 Atari's Video Computer System comes complete with two joysticks, mains adaptor and a PacMan cartridge. Being 'dedicated' to game playing, the VCS cannot be used as a general-applications computer, and all games are cartridge-based

communicate, via telephone lines, with a database of games software. In addition it was also an EPROM burner with slots to accept the edge connectors of the major cartridges. The significance of this was that it was now possible to go to a local dealer, get an instant readout of the Top Twenty best selling games, and select one for immediate burning-in on a Romox blank cartridge.

An alternative to this method of distribution was Gameline, the 'pay-as-you-play' system set up in the US by Bill van Meister. Gameline markets a plug-in modem for Atari VCS machines that attaches the home computer to the telephone system. Games on this system originally cost one dollar for 45 minutes of playing time.

Major Havoc
During most of the 1970s Atari rode high on the profits generated by arcade machines like this, one of the many 'zapping' games. The advent of the home computer required a completely new marketing strategy

Two of the biggest personal computing networks in the US, Compuserve and The Source, offer games software as part of their regular service to suppliers who likewise plug personal computers into a remote database via a modem and the telephone network. Coleco, a manufacturer of home video games, has linked with AT&T (American Telephone and Telegraph) to supply an interactive entertainment service. Atari sees this as the way forward and has linked with Activision to send games programs over the telephone network in its secret Ataritel project. So long as Atari are in league with Warner Communications, this is a good idea, but if Atari is sold off then it loses access to Warner Amex Cable Communications, on which the scheme depends.

Atari has had its problems, caused partly in the past by strife between the fading videogame division and the rising home computer division of the company, and now resolved by their integration. But it still has a good range of home machines that have long led the way in home computer graphics and easy-to-use software. The newest home machines are similar in design to the old ones, which were notably in advance of their time. These machines feature three custom chips: Pokey, Antic and GTIA, controlling respectively the Input/Output ports, graphics and colour.

All the machines are based on the 6502 processor and a useful variety of utility programs are now available. These include: VisiCalc, Atariwriter (a word processing package) and a home management program. The Z80 softcard is now a reality, and this makes the Atari computers suitable for equipping with Digital Research's Personal CP/M.

The company is also turning its attentions to software. Its UK division has appointed a software troubleshooter to handle all software problems, to market its best software suitably adapted for other micros (notably the Commodore machines) and, most significantly, to watch the British scene closely for rising young games programmers. After a couple of years in the doldrums, Atari has the potential for a resurgence to its great days.

A dream computer, A dream holiday, and a sight for twenty sore eyes.

WIN – 1st Prize, an I.B.M. P.C. system to the value of £5,000.

WIN – 2nd Prize, a holiday for two in the U.S.A. with a visit to Silicon Valley.

WIN – 3rd Prize, one of ten high resolution colour monitors.

ALL YOU HAVE TO DO

"AH! DRESS THE APE RIGHT," EXCLAIMED AMOS, BUGLE SALES MANAGER.

This crazy sentence contains anagrams of four familiar pieces of computer jargon.

Each week until issue 4 we will be featuring the particular parts of the above sentence that make up one of the four words or phrases.

This week (issue 2), the well-known computer word or phrase is hidden in the words "RIGHT AMOS"

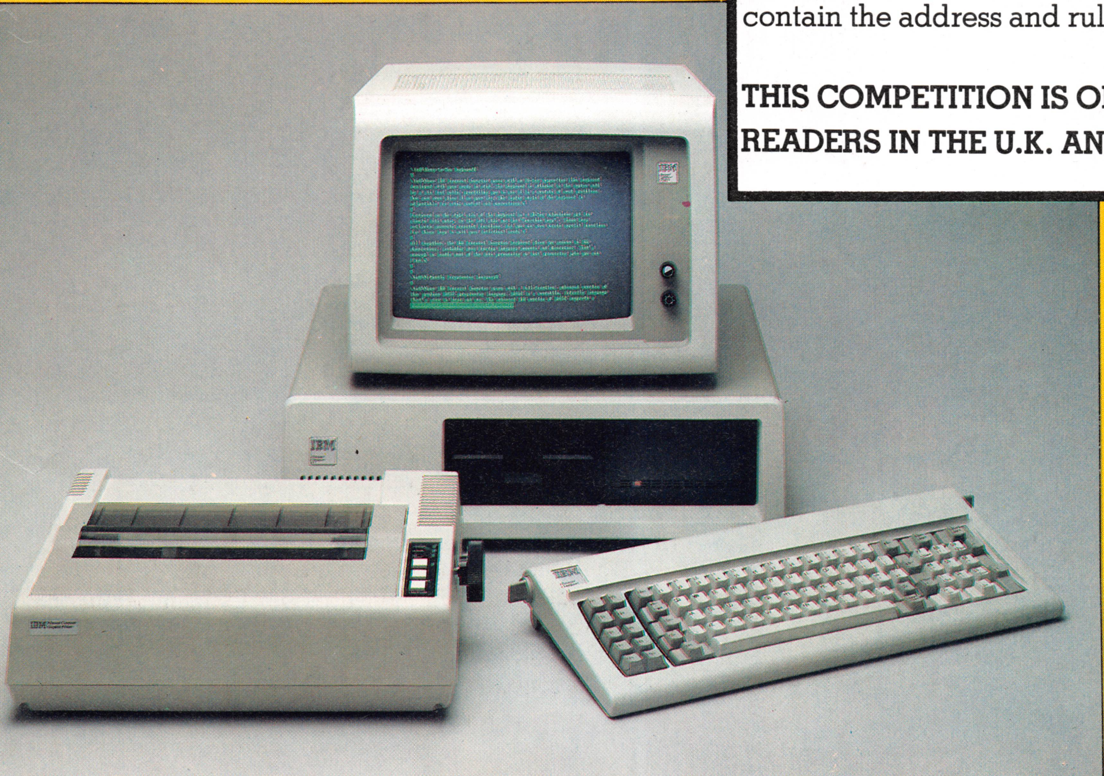
HOW TO ENTER

To enter the competition, first you must correctly identify each week until issue 4 all four hidden words or phrases. Then use your skill and knowledge of computing *and* your sense of humour to invent a new piece of computer jargon, no more than 5 words long, and give its definition in no more than 15 words.

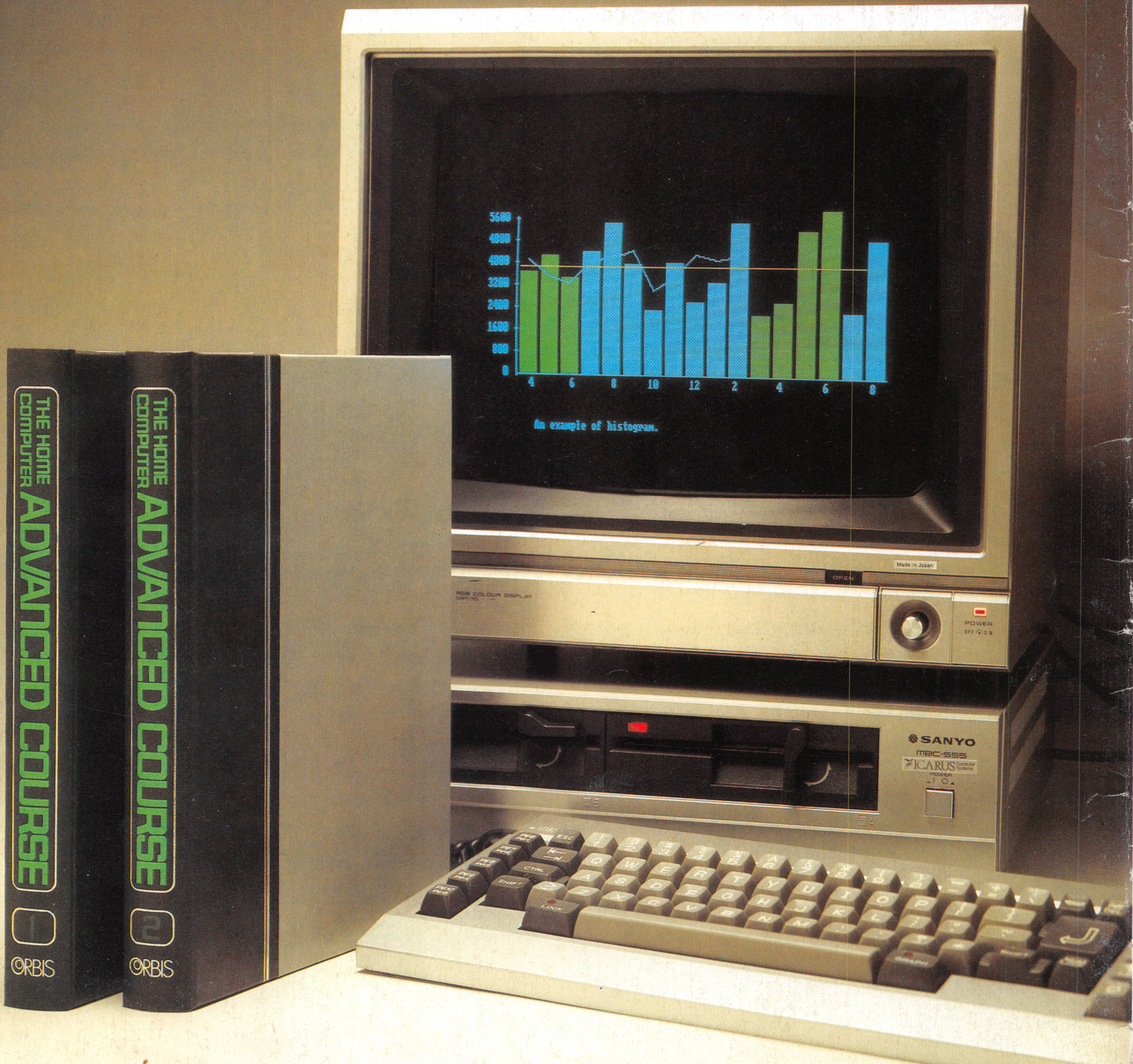
REMEMBER

Retain your answers to issues 1, 2, 3 and 4. Issue 4 will contain the address and rules for competition entries.

THIS COMPETITION IS OPEN TO READERS IN THE U.K. AND EIRE ONLY



The volume 1 binder can be yours free!



THE HOME COMPUTER ADVANCED COURSE 1 ORBIS
THE HOME COMPUTER ADVANCED COURSE 2 ORBIS
An example of histogram.
SANYO
MBC-555
ICARUS

The Home Computer Advanced Course will take you far beyond the novice stage, widening your knowledge and making you a more sophisticated user.

To help you keep your copies immaculate, we will be making a very special **free binder offer** in Issue 5 – be sure not to miss it!

Overseas readers: this special offer applies to readers in the U.K., Eire and Australia only. Binders may be subject to import duty and/or local tax.